2003

Formation of flexible manufacturing cells with human lifting consideration.

Setiadi. Lesmana
University of Windsor

FORMATION OF FLEXIBLE MANUFACTURING CELLS WITH HUMAN LIFTING CONSIDERATION

by

Setiadi Lesmana

A Thesis submitted to the
Faculty of Graduate Studies and Research
through the Department of Industrial and Manufacturing Systems Engineering
in partial fulfillment of the requirements for
the degree of Master of Applied Science
at the University of Windsor

Windsor, Ontario, Canada

2002

Your file Votre référence
ISBN: 0-612-84545-1
Our file Notre référence
ISBN: 0-612-84545-1

Abstract

Global economic growth and international trade have forced the manufacturing companies to face the significant increase in level of global competition. Customers are now requiring custom-made products with high quality and variety at a reasonable competitive price. The manufacturing companies are being forced to design and organize their manufacturing operations to survive the competition and to satisfy the customers' needs. Cellular Manufacturing Systems (CMS) have been considered as an important step in achieving these objectives.

In this research, mixed integer programming models are developed. The objective of the models is to minimize the total cost function associated with the formation of the manufacturing cells, while considering the human limitation of manual lifting tasks. In addition, the multi period models consider machine relocation and/or manpower level change to balance machine and operator workload throughout the production periods. The models consider machine procurement cost, machine idle time cost, intercellular movements cost, operator cost, operator idle time cost, manual materials lifting risk penalty, machine relocation cost, and operator relocation cost. The manual materials lifting risk penalty in the models is based on the revised NIOSH lifting equation. Higher lifting index can be associated with higher injury risk for the operators.

The mathematical models are tested using three methods: sequentially using combination of Branch-and-Bound and Heuristic (SeqBBH), simultaneously using combination of Branch-and-Bound and Heuristic (SimBBH), and simultaneously using Genetic Algorithm (SimGA). The SeqBBH method consists of 2 steps: (1) solving the part families and machine cell formations problem, and (2) solving the operator assignment. The SimBBH method and SimGA method solve the models simultaneously for part families, machine cell formations, and operator assignment. The Genetic Algorithms are developed to deal with the nonlinearity in the models and to reduce the computational time.

The test results from each method using eight numerical examples are then analyzed and compared in terms of the system cost value (objective function) and computational time. Finally, the consistency of GAs is also tested.

Love and Dedication

To My Family

I hereby declare that I am the sole author of this thesis. I authorize the University of Windsor to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Setiadi Lesmana

I further authorize the University of Windsor to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Setiadi Lesmana

Acknowledgements

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Salem Taboun, who has given me this research topic, ideas, advice, constant encouragement, and support during my course work and my research to complete this thesis.

I would like to express my gratitude towards my committee members, Dr. Michael Wang, for his continuous advice and support over the past year, and Dr. Waguih ElMaraghy, for his advice and support during the last month of my thesis manuscript preparation.

I would like to thank my friends, Mr. Ram Barakat and Ms. Jacquie Mummery, for their friendship and support since the first day I joined this department in May 2001. I would like to extend my thanks to my friend, Navneet Vidyarthi, for his help and suggestion in my research.

My special thanks and truly respect to Mr. and Mrs. McCready, who made me feel like their family since the first day I arrived in Windsor. Also thanks to Dr. Wasi Pramono and family for their help and support.

I would like to thank my friends in Canada for their friendship and help during my study. Also special thanks to my friends in Indonesia for their friendship and constantly support despite the distance.

Finally, I would like to give my special thanks, love and dedication to my grandmother, my parents, and my brothers and sister.

This research was supported by Dr. Taboun's grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1 – Introduction

1.1. General Overview

Henry Ford once stated, "People can order any color as long as it is black". At that time, markets were regional and there was no foreign competition. The orientation of production systems has changed from the producers' point of view to the customers' point of view. Today's market is determined by customers. For producers to exist, they must seek and produce what potential customers require.

Several decades of global economic growth and international trade have been forcing the manufacturing companies around the world, especially in North America, to face the significant increase in level of global competition. The customer's sense of value has been gaining variety throughout the decades. Once satisfied with the low quality and low variety of products, customer's taste is now requiring more customization to individual likes and dislikes. As a result of these changes, the manufacturing companies, which were ignorant to customer's needs, are being forced to design and reorganize their manufacturing operations, so that they can produce the custom-made products in a short period of time with high quality and variety and at a reasonable competitive price (Ozdemir 1995).

In order to respond to these challenges and to survive the competition, the manufacturers have attempted to implement the steps necessary to achieve the factory of the future, which can easily compete in the future's global economy. Cellular

Manufacturing Systems (CMS) have been considered as an important step in achieving these objectives (Black 1991). CMS is an application of the Group Technology (GT) concept to factory reconfiguration and shop floor layout design. It involves processing a collection of similar parts on a dedicated group of machines or manufacturing processes (Irani *et al.* 1999). Each cell is dedicated to processing a specific set of part families. A manufacturing cell typically consists of several functionally dissimilar machines, whereas a part family consists of a set of parts with similar processing requirements (Askin *et al.* 1997).

GT has proven to be very successful when implemented properly. Prior studies (Pullen 1976; Houtzeel and Brown 1984; Wemmerlöv and Hyer 1989, and Wemmerlöv and Johnson 1997) have shown the following dramatic improvements:

- throughput time (5–90%),
- work-in-process inventory (8–80%),
- materials handling (10–83%),
- job satisfaction (15–50%),
- fixtures (10–85%),
- setup time (2–95%),
- space needed (1–85%),
- quality (5–90%),
- finished goods (10–75%).

The advantages above essentially lower manufacturing cost and produce a higher quality product. This is what makes GT so attractive. GT is intended for medium variety

and medium volume production environments. However, manufacturing cells can be used whenever short sequences of processing steps are found with sufficient demand volume to justified dedicated equipment.

The fundamental problem in CMS is to design the manufacturing cells that consist of part families and machine cells to increase the efficiency of the system, for example, to avoid or remove the parts moving between cells (intercellular movements). Without these exceptional elements, the independent manufacturing cells will simplify the scheduling operations which will improve the shop floor control and also simplify the material flow. However, forming the manufacturing systems into independent cells is a hard goal to attain because of the existence of the bottleneck elements or bottleneck machines, especially for large-scale problems. A bottleneck machine is the one which is needed in more than one cell, whereas a bottleneck element is the one which needs machines in more than one cell. There are several approaches dealing with bottleneck elements in the literature, which can be stated as follows (Ozdemir 1995):

- redesign the part, so that it does not require the bottleneck machine,
- try to allocate the operation of the part, which requires the bottleneck machine within the cell,
- duplicate the bottleneck machine,
- subcontract the manufacture of the part.

The decision to duplicate the machines incurs additional investment costs, as well as idle time cost unless the new duplicated machine can be fully utilized. The idle time cost on the machine represents the loss of interest on the capital invested. This alternative must be evaluated by taking into account both financial, such as machine investment and

idle time costs, and non-financial considerations, such as simplified material flow and job floor control. The decision to subcontract a part, also involves financial and non-financial consideration. Financial considerations involve the evaluation of the difference between the fixed and variable costs of manufacturing and the purchase cost of buying the part from outside vendor. Objectives of the company should be incorporated into these financial analyses, for example, if the objective is to produce high quality products and maintain Just-In-Time (JIT) delivery, and if the supplier is not able to satisfy these objectives then the financial analyst should assign higher subcontracting costs to those particular parts. Also the subcontracting costs should reflect the effect of simplified production flow by subcontracting. Non-financial consideration may involve the loss of employee morale and motivation, because large-scale subcontracting can have much the same impact on employees as plant closing or relocation of plants (Kumar and Vanelli 1987).

Most of the previous research in the existing literature also indicated that human issues in the GT have not been studied and developed significantly. One of the important aspects in the human issues is the worker assignment to the machine cell. Workers or operators can be assigned to the manufacturing cells using several criterions, for example: technical skills, expertise, or ergonomics concern.

The research in this thesis is focused on developing mathematical models for design of manufacturing cells with worker assignment based on ergonomics point-of-view.

1.2. Objectives of the Research

The objectives of the research are as follows:

i. To develop mathematical models to simultaneously solve the part family and machine cell formation problems with incorporating the human limitations in the objective function and constraints.

ii. To develop the models above into multi-period planning horizon.

iii. To develop a heuristic technique to solve the mathematical models and compare the results in terms of the objective cost functions and computational time consumed.

1.3. Organization of the Research

The research in this thesis is organized as follows:

Chapter 2: Literature review on various aspects of part family and machine cell formations in cellular manufacturing systems and a summary of human limitations in manual-material-handling activities.

Chapter 3: Development and explanation of the mathematical models.

Chapter 4: Development of heuristic techniques.

Chapter 5: Numerical examples to test the mathematical models.

Chapter 6: Conclusion and recommendation for future research.

Chapter 2 – Literature Review

This chapter is divided into two sections; the first section is the literature review on the cellular manufacturing systems, and the second section is on the human limitation associated with manual-material-handling systems.

2.1. Cellular Manufacturing Systems

2.1.1. Definition of Cellular Manufacturing

Some definitions of Cellular Manufacturing (CM) are described in the following representation.

Askin *et al.* (1997): "Cellular manufacturing is an application of group technology, entails the creation and operation of manufacturing cells. Each cell is dedicated to processing a specific set of part families. A manufacturing cell typically consists of several functionally dissimilar machines, whereas a part family consists of a set of parts with similar processing requirements".

Selim *et al.* (1998): "Cellular manufacturing is an application of the Group Technology philosophy in manufacturing. Group technology itself can be defined as a manufacturing philosophy identifying similar parts and grouping them together into families to take advantage of their similarities in manufacturing and design. Cellular manufacturing is concerned with the creation and operation of manufacturing cells which are dedicated to the production of a set of part families".

Ramabhatta and Nagi (1998): "Cellular manufacturing is an application of Group Technology, a philosophy whose main idea is to capitalize on similar, recurrent activities, to increase profitability and overall operational efficiency of a manufacturing organization, typically a batch manufacturing system. Cellular manufacturing involves processing collections of similar parts (part families) on dedicated clusters of dissimilar machines or manufacturing processes (cells)".

Irani *et al.* (1999): "Cellular manufacturing is an application of the Group Technology concept to factory reconfiguration and shop floor layout design. Cellular manufacturing involves processing a collection of similar parts on a dedicated group of machines or manufacturing processes".

Molleman *et al.* (2002): "A cellular manufacturing system is defined as the grouping of workers and machines into relatively independent cells, which are responsible for the complete manufacturing of a set of part types".

2.1.2. Cell Formation

One of the first problems in implementing cellular manufacturing is the Cell Formation. Cell Formation (CF) deals with the identification of the family of parts and the group of machines on which these parts are to be processed. Wu and Salvendy (1993) stated that the CF problem may be defined as: "If the number, types, and capacities of production machines, the number and types of parts to be manufactured, and the routing plans and machine standards for each part are known, which machines and their associated parts should be grouped together to form cells?".

CF assumes that a set of parts is identified as suitable for manufacture on a specified group of specific machines or machine types. To do this, there must be a basic relationship between a part and a set of machines (e.g., a part routing). Parts can then be assigned to families such that all parts in the family are processed on the same group of machines, and similar machines can be grouped into cells if they process the same set of parts. Most procedures for CF rely on this type of relationship to establish part families and machine cells. Once the part and machine populations for CM have been identified, the CF problem can be reduced to three major decisions:

(a) identification of part families;

(b) identification of machine cells; and

(c) allocation of the families to cells or vice versa.

These three decisions are interrelated and compose subproblems of the CF problem (Selim *et al.* 1998).

2.1.3. Cell Formation Solution Methods

Several CF solution methods and techniques have been developed by several researchers. The methods and techniques can be classified into 6 main groups:

1. Classification and Coding Approach,
2. Algorithmic Approach,
3. Heuristic Approach,
4. Graph Partitioning Approach,
5. Artificial Intelligence Approach, and
6. Mathematical Programming Approach.

The review of literature on CF solution methods and techniques is presented in the following sections.

2.1.4. Classification and Coding Approach

Tatikonda and Wemmerlöv (1992) reported an empirical study of classification and coding (CC) system usage among six manufacturing firms. The case studies included investigation, selection, justification, implementation, and operation of CC systems by the manufacturers. User characteristics and experiences were compared and analyzed across seven cases.

2.1.5. Algorithmic Approach

Several algorithmic approaches have been developed for part family and machine grouping problems. These approaches then can be divided into 3 main sub categories: array-based clustering approach or machine-component manipulation approach, similarity coefficient approach, and other algorithmic approach.

2.1.5.1. Array-Based Clustering or Machine-Component Manipulation Approach

The main objective of array-based clustering or machine-component manipulation approach is to divide the components into families and machines into groups, in such a way that each component can be fully processed in one group (Ozdemir 1995). In this approach, the processing requirements of the components on machines can be represented by an incidence matrix. This is referred to as the machine-component matrix $\tilde{A}$. The machine-component matrix has zero and one entries (a_{ij}). A "1" entry in row i and

column j (a_{ij} = 1) of the matrix indicates that component j has an operation on machine i, whereas a "0" entry indicates that it does not. The array-based techniques try to allocate machines to groups and components to associated families by appropriately rearranging the order of rows and columns to find a block diagonal form of the a_{ij} = 1 entries in the machine-component matrix.

Production Flow Analysis

Burbidge (1963) proposed one of the earliest algorithmic approaches for the CF problem which is referred to as Production Flow Analysis (PFA). PFA is a technique to divide all the parts completely into part families and to divide all the existing machines completely into associated groups. It is based on the progressive analysis of the information contained in the route cards for the components and assemblies produced in factory by concentrating on the methods of manufacturing without attempting to change them. A manual method for CF called Nuclear Synthesis is proposed where manufacturing cells are created around 'key machines'.

Component Flow Analysis

El-Essawy and Torrance (1972) proposed a method called Component Flow Analysis (CFA). In some respects, the methodology of CFA differs from that of Burbidge's PFA procedure since the latter first partitions the problem, whereas the former does not.

Bond Energy Algorithm

McCormick *et al.* (1972) developed a method called the Bond Energy Algorithm. This algorithm involves the evaluation of so called 'bond energy' in the part-machine matrix. A bond is said to exist between a pair of adjacent row elements or column elements if the pair of elements both have nonzero values. The value of the bond is equal to the product of the two adjacent elements. The total bond energy of the matrix is equal to summation of the product of any two adjacent elements. The algorithm manipulates the columns and rows of the part-machine matrix and tries to find a matrix containing the highest total bond energy. This algorithm can identify part families and machine cells simultaneously but still needs extensive manipulation of the final part-machine matrix to form cells of the required size.

Rank Order Clustering Algorithms

King (1980) developed the Rank Order Clustering (ROC) algorithm which rearranges the rows and columns of the initial machine incidence matrix in decreasing binary values to obtain a block diagonal form. However, the applicability of the algorithm was restricted by the strong dependence of the results on the initial order of machine-part matrix and existence of storage problems created by the usage of binary value used for reallocation.

King and Nakornchai (1982) implemented the ROC2 algorithm, a more efficient version of the previously published ROC algorithm. And it was described together with a new relaxation procedure for bottle-neck machines. In the part-machine matrix, four initial cells are formed and several rows or columns can be sorted at the same time

instead of element by element. In addition, an even faster sorting procedure called least significant digit radix sort is used to further improve the efficiency.

Even though ROC2 algorithm had been developed, there are still some drawbacks. Chandrasekaran and Rajagopalan (1986b) pointed out that the solution generated by ROC2 algorithms is highly dependent on the initial matrix. This problem is often found in clustering techniques. Different initial seeds will lead to local optimal points instead of a global optimal point. Nevertheless, the ROC and ROC2 algorithms are the most widely used algorithm for the cell formation problems (Chen 1995).

Direct Clustering Algorithm

Chan and Milner (1982) developed the Direct Clustering Algorithm (DCA) to solve the part family and machine grouping problems for cellular manufacturing systems. The DCA algorithm includes 4 steps: (1) count the number of positive entries in each row and column of the part-machine matrix, (2) starting from the first column, transfer the rows with positive entries in that column to the top portion of the matrix, (3) starting from the first row, transfer the columns with positive entries in that row to the left-most portion of the matrix, (4) iterate between steps (2) and (3) until no further transfer is required. The procedure also allows user interaction to deal with the problems of bottlenecks and exceptional elements when they occur.

Modified Rank Order Clustering

Chandrasekharan and Rajagopalan (1986b) developed an extension of the Rank Order Clustering algorithm called MODROC. The MODROC algorithm is a combination

of the ROC algorithm and the similarity coefficient technique to overcome the problems of the ROC algorithm.

There are three main stages in the MODROC algorithm. In the first stage, the ROC algorithm is applied on the rows and columns of the part-machine matrix for two iterations. The resulting matrix will include a block in the upper-left corner. A block is defined as a maximum submatrix which contains only 1's in the part-machine matrix.

In the second stage, the block in the upper-left corner is identified and represents a primary cell with corresponding part family and machine group. The rows (parts) corresponding to the block are sliced off and removed from further consideration. ROC algorithm is then applied to the truncated matrix again followed by the block and slice-off procedures. This process is repeated until all parts are grouped in blocks (cells). Note that, from the results of stage 2, the part families are mutually exclusive with each other but the machine groups are not necessarily non-intersecting.

In the third stage, a similarity measure is used to compare the similarity between each primary cell. The similarity measure is the ratio of number of common machines required by each pair of part families of the corresponding cells to the smaller total number of machines of the two cells. Then the algorithm finds the pair of primary cells with highest measure and joins the two cells into one. Stage three is repeated until the similarity measure of each pair of part families of the corresponding cells is equal to 0 (i.e., all machines are disjoint). If this does not happen, the process is repeated until number of cells is equal to 1. The final cluster formed is the solution of the cell formation problem.

Comparison of ROC – DCA – BEA

Chu and Tsai (1990) examined and compared three array-based clustering algorithms for cell formation: rank order clustering (ROC), direct clustering analysis (DCA), and bond energy analysis (BEA). Several problems were tested by using these algorithms and the results were compared by using four different criteria: total bond energy, percentage of exceptional elements, machine utilization, and grouping efficiency. It was concluded that the bond energy analysis (BEA) produced better results than the other two methods.

Intercell-and-Intracell-Moves-based Algorithm

Chow and Hawaleshka (1993) developed an algorithm to solve the machine grouping problem that minimize the intercellular movements with allocating a new machine. It is observed that the total number of exceptional parts generated by the $(n+1)$ total number of machine cells is always greater than those generated by n total number of machine cells.

2.1.5.2. Similarity Coefficient Approach

Several researchers developed techniques to form the part families and machine cells based on similarity coefficients. The similarity measures are generally based on the sequence of operations, the processing requirements of parts, the tooling requirements of parts and availability of the tools on the machines, the production volume, and the unit operation times of parts.

The First Similarity Coefficient

McAuley (1972) was the first to suggest the use of similarity coefficient techniques in cell formation problems. A similarity coefficient was proposed based on the single linkage cluster analysis. It is a hierarchical process of machine grouping done according to the computed similarity coefficient. The similarity coefficient defined as 'the number of components which visit both machines, divided by the sum of components which visit one or other of the machines'. A major disadvantage of this approach is that it may cause the chaining problem. The chaining problem is referred to the situation when two machines have very high similarity measure and one of them has been included in a machine group. Then, the other one will be automatically included in that machine group even though the similarity measure between the new included machine and the rest of machines in that group is very low.

Three Similarity Coefficients

de Witte (1980) stated that in PFA, routings are the basic data from which interrelations between operations in a functionally structured production system may be found. One possibility of describing interrelations between operations is by using similarity coefficients. Machine types were grouped into three categories: primary machine types, secondary machine types, and tertiary machine types. Primary machine types are the machine types that have only one unit available or all units of that machine type need to be grouped into one cell. Secondary machine types are the types of machines which must be duplicated in several cells. Tertiary machine types are those that can be duplicated if desired.

Three new similarity measures were defined based on the machines types mentioned before. The first one shows the 'absolute relations' between machine types (SA). The second one shows the 'relative mutual interdependence' between machine types (SM). The third one shows the 'relative single interdependence' between machine types (SS). The three similarity coefficients were used in the 4-step method to analyze the relations between machine types and allocate machine types to cells. However this approach only identifies machine groups but not part families.

MACE: Machine-Component Cell Formation

Waghodekar and Sahu (1984) developed a method to minimize the number of bottleneck parts for cell formation problems based on the similarity coefficient of the product types. The method called MACE (machine-component cell formation) which used three types of similarity coefficient: similarity coefficient of additive type, similarity coefficient of product type based on the total number of components processed by each pair of machines, and similarity coefficient based on total flow of common components processed by a particular machine type. The procedure produced consistent results irrespective of the sequence of machines and the parts in the initial machine component matrix.

Average Similarity Coefficient

Seifoddini and Wolfe (1986) used the average linkage algorithm to overcome the chaining problem caused in single linkage algorithm. In average linkage algorithm, the similarity coefficient between two clusters is defined as the average of similarity

coefficients between all elements of the two clusters. In addition, they employed a bit-level data storage technique which they claimed reduced storage requirement and computational effort. They also presented a procedure to identify exceptional elements and to duplicate bottleneck machines.

Ideal Seed Non-Hierarchical Clustering and ZODIAC Algorithm

Chandrasekharan and Rajagopalan (1986a) proposed an ideal seed non-hierarchical clustering (ISNC) algorithm which involves three primary stages. In the first stage, the problem is formulated as a bi-partite graph which consists of a machine subgraph and a part subgraph. The *k*-means from Anderberg (1973) is adapted to construct k parts and k machines by grouping vectors which are close together. In the second stage, a performance measure called group efficiency is used to compare different grouping alternatives. In the third stage, parts and machines are rearranged to the closest 'imaginary groups' in an attempt to improve the initial assignment.

Later, Chandrasekharan and Rajagopalan (1987) presented an improved version – ZODIAC algorithm. ZODIAC stands for zero-one data: ideal seed algorithm for clustering. It uses the concept of natural seeding instead of the 'ideal' seed to improve the solution quality. The formation of part families and machine cells has been treated as a problem of block diagonalization of the zero-one matrix. Different methods of choosing seeds have been developed and tested.

Commonality-Score-based Clustering Algorithm

Wei and Kern (1989) presented a linear clustering algorithm. The algorithm is based on the calculation of a commonality score which indicated the relative value of having two machines in the same cell (i.e., expression of the similarity of the sets of parts on which the two machines work). The algorithm considers the limitation on the cell size and the number of cells. It is flexible in the sense that various evaluation criteria can be easily incorporated.

Production-Data-based Similarity Coefficient

Gupta and Seifoddini (1990) presented a similarity coefficient based approach to the problem of machine-component grouping. Relevant production data, such as part type, production volume, routing sequence, and unit operation time are incorporated in the early stages of grouping decision. The algorithm developed also suggests a methodology for evaluating alternative solutions from different algorithms on a quantitative basis using a modified version of an existing coefficient. The algorithm then identifies bottleneck machines and corresponding cell candidates for their duplication using percentage utilization in each cell as a criterion. However, the algorithm ignores the cost of duplicating the bottleneck machines.

Similarity Coefficient with Ill-Structured Matrix Consideration

Kusiak and Cho (1992) presented two similarity measures of the group technology problem. The first one is applicable to the cell formation problems with consideration of alternative routings. The second one generalizes the first so as to work

well for ill-structured matrix. An ill-structured matrix is a non-decomposable matrix that tends to have many exceptional elements in the final matrix formation. Two heuristic algorithms were used and the results show that similarity coefficients have a significant effect on the solution quality.

Constrained Hierarchical Clustering

Shiko (1992) presented the application of constrained hierarchical clustering to similarity analysis, the grouping of process plans and the parts manufactured in accordance with them. It proposed solutions to some important problems, such as determining a metric of process plan similarity evaluation, the study of a cluster-creating technique with two process plans and a rule for singling out the standard plan, the determination of a logical threshold value for partitioning into families.

Machine-Level-Based Similarity Coefficient

Süer and Ortega (1994) developed a method to form manufacturing cells based on machine level based similarity coefficient. The similarity coefficient takes into account the unit processing times of parts and the number of machines required from each machine type. However, the sequences of operation and production volumes of parts were not considered.

Multiple Criteria Clustering Algorithm

Won and Kim (1997) developed an approach to cell formation based on machine similarity coefficient. This approach saves considerable computer memory required to

store the similarity coefficient information in comparison with the methods using the similarity coefficient defined between parts. Furthermore, the new definition includes existing machine similarity coefficient as a special case. Unlike the existing algorithms using single clustering criterion, a new algorithm using multiple clustering criteria is developed. The algorithm using the generalized machine similarity coefficient effectively solves large-sized and ill-structured problems.

Multivariate Analysis Model

Kitaoka *et al.* (1999) formulated a multivariate analysis model to generate optimal machine cells and part families in group technology problems. The algorithm is carried out in three stages. The double centering matrix for similarity of machines and parts is used as similarity coefficient matrix. A quantification method is applied to find the eigenvalues and eigenvectors on the double centering matrix. Cluster analysis is applied to make part families and machines groups while minimizing the distance of eigenvectors.

Survey, Evaluation, and Comparison of Similarity Measures

Taboun *et al.* (1991) used computer simulation to compare and evaluate three grouping criteria used to cluster parts and machines into part families and machine cells. Three types of similarity measures are used for the grouping criteria: machine similarity, process similarity, and tooling similarity. Three grouping criteria are simulated using SIMAN and an experimental design is conducted to observe and evaluate their performance of handling stochastic changes in the part mix and production volume levels

in terms of two evaluation criteria: parts produced and the makespan. They concluded that grouping derived from machine similarity was superior in terms of both evaluation criteria.

Shafer and Rogers (1993) presented a survey on similarity and distance measures in cellular manufacturing systems, and developed a new similarity measure which eliminates the biases that resulted from the differences in number of parts a machine processes or from the differences in the number of machines a part requires in their processes.

Seifoddini and Hsu (1994) presented a comparative study of similarity coefficients and clustering algorithms for machine cell formation. Three different similarity coefficients are compared by solving various machine component grouping problems. The similarity coefficients used are the Jaccard's similarity coefficient, weighted similarity coefficient, and commonality score. The results from the study stated that weight similarity coefficient generated better solutions based on the number of exceptional parts.

2.1.5.3. Other Algorithmic Approach

Logendran and West (1990), and Logendran (1990) developed an algorithm for determining optimal / near optimal machine-part clusters in cellular manufacturing. The algorithm based on a model describing the total moves contributed by both intercellular and intracellular moves. Appropriate weights have been chosen to represent the total moves in equation form as a weighted sum of both intercellular and intracellular moves. In addition to the total moves, utilization of a workstation in a cell had been evaluated

and used in the determination of the best workstation and part assignments to the respective cells from among a choice of solutions.

Logendran (1991) revised the previous model developed (Logendran and West 1990, and Logendran 1990) with a model that involves two important factors: first, the sequence of operations in evaluating intercellular and intracellular moves, and second, the impact of the layout of cells in evaluating the intercellular moves. The total moves was computed as a weighted sum of both intercellular and intracellular moves, and was used to measure the performance / fitness of the model. An algorithm for the model was then developed, implemented, and tested on different test problems based on the literature.

Beaulieu *et al.* (1997) developed an algorithm to solve the problem in designing cellular manufacturing systems in an acceptable time while considering machine capacity, alternative routing, and constraints on cell size. The algorithm is composed in two main phases. The first phase forms a configuration of independent cells using a cell aggregation procedure. The second phase addresses the problem of low machine utilization.

2.1.6. Heuristic Approach

The Occupancy Value Method

Khator and Irani (1987) presented an approach using heuristic procedure called the Occupancy Value method for identifying clusters in a machine-component matrix created from route card data. This method groups the parts with the highest occupancy value into part groups by progressively developing block diagonalization of the incidence

matrix, and uses the part-machine incidence matrix only as input instead of using it for row-column manipulations (array-based clustering technique). However, the method had difficulties dealing with exceptional elements.

Heuristic with Economic consideration

Askin and Subramanian (1987) proposed a heuristic approach to the economic determination of machine groups and their corresponding families for GT. The procedure considers costs of work-in-process and cycle inventory, intra-group material handling, set-up, variable processing and fixed machine costs. The three stage procedure initially reorders part types based on routing similarity. Then try to combine adjacent part types to reduce machine requirements. Finally, groups are combined where economic benefits of utilization offset those of set-up, work-in-process and material handling.

Heuristic for Part-Family / Machine-Group Problem (PF/MG)

Ballakur and Steudel (1987) presented a heuristic to the problems in the cell formation. The heuristic considers several practical criteria such as within-cell machine utilization, work load fractions, maximum number of machines that are assigned to a cell, and the percentage of operations of parts completed within a single cell. An application of the heuristic to a large sample of industrial data involving 45 worksites composed of 64 machines, and 305 parts was given.

Okogbaa *et al.* (1992) developed a heuristic for part families and machine grouping (PF/MG) problem. The algorithm operates in three stages. In the first stage, machines are grouped into cells. In the second stage, the algorithm rearranges the

machine cell configurations in order to minimize the inter machine flow. In the third stage, each part is assigned to the machine cell that performs the maximum number of its operations. Model simulation of this heuristic was also developed and compared to the simulation of a traditional process layout (TPL) system based on three performance criteria: machine utilization, queue length, and flow time.

Klippel *et al.* (1999) developed a two-phase heuristic procedure for cell formation in cellular manufacturing systems. The first phase consists of a version the breadth-first search algorithm which allocates parts to the machine in a way not to violate the processing capacity of the machines. And the second phase consists of a genetic algorithm which identifies the machine cells and the respective part families.

Mukattash *et al.* (2002) developed three heuristic procedures to deal with the complex and multi-criteria cell formation problems. The heuristics are designed to assign parts to the cells in presence of alternative process plans, multiple alternative (parallel) machines, or when processing times are taken into consideration.

Heuristic for Eliminating Exceptional Elements

Kern and Wei (1991) developed a method to minimize the exceptional elements or to minimize the intercellular movements, after the initial cell formation has been obtained. The method analyzes each exceptional element and calculates the most effective sequence to remove exceptional elements.

Flexible Manufacturing Cells Design Heuristic with Alternative Routing Considerations

Gupta (1993) developed a heuristic procedure to design flexible manufacturing cells with alternative routing considerations. A similarity index was developed based on alternative routing sequences, capacities of the machines, production volumes, and operation times. The alternative routing sequences for each product is developed using a manufacturing planning software (MANUPLAN). Part families and machine cells are developed by using Complete Linkage (CLINK) as the clustering technique, such that each part is assigned to the cell that performs the maximum amount of its total operations. However, the scheduling constraint was not considered in this research.

Heuristics Branching Rules

Cheng *et al.* (2001) evaluated several branching rules for a branch-and-bound algorithm for solving the group technology problem. Furthermore, a branching rule to deal with the general GT problem in which consider exceptional machines is also developed.

Simulated Annealing, Tabu Search, and Genetic Algorithm Heuristics

Vakharia and Chang (1997) developed two heuristic methods for generating solutions to the group technology problem. These methods are based on two powerful combinatorial search methods: simulated annealing and tabu search. The performance of the heuristics is examined using randomly generated, published and industry data. The results indicate that the simulated annealing based heuristic is the preferred technique in the context of the problem addressed in the paper. Furthermore, the simulated annealing

based heuristic is also demonstrated to give a near-optimal solution to the cell formation model formulated.

Onwubolu and Mutingi (2001) developed a genetic algorithm (GA) metaheuristic-based cell formation procedure. The cell formation problem is to simultaneously group machines and part-families into cells so that intercellular movements are minimized. An option for considering the minimization of cell load variation is included and another option, which combines minimization of intercellular movements and cell load-variation, is also included. The algorithm solves this problem through improving a cell configuration using the GA metaheuristic. The designer is allowed to specify the number of cells required a priori and impose lower and upper bounds on cell size. This makes the GA scheme flexible for solving the cell formation problems.

Distributed Dynamic Grouping Methodology

Ben-Arieh (1998), and Ben-Arieh and Sreenivasan (1999) presented the distributed dynamic grouping methodology that allows parts to be grouped as they arrive. This method is developed to overcome with the contemporary manufacturing that advocates flexibility with less commitment to the features used for clustering. The algorithm is based on negotiations among agents, who accept parts into their groups, and then the algorithm is implemented in C++ language.

2.1.7. Graph Partitioning Approach

Graph partitioning methods treat the machine and/or parts as vertices and the processing of parts are the arcs connecting these nodes. These models aim at obtaining

disconnected subgraphs from a machine-machine or machine-part graph to identify manufacturing cells.

Rajagopalan and Batra (1975) presented a graph partitioning approach to form the machine cells in group technology. Input data derived from the route cards of the components is analyzed and used to derive a graph whose vertices correspond to the machines and whose edges represents the relationship created between machines by the components using them. After the creation of cells by using the graph partitioning approach, the parts are allocated to the cells and the number of machines of a particular type in each cell is determined.

Askin and Chiu (1990) proposed a heuristic graph partitioning procedure for the machine assignment and cell formation. First, a mathematical programming model is developed to incorporate costs of inventory, machine depreciation, machine setup, and material handling. The formulation is then divided into two phase/subproblems: the first subproblem assigned components to specific machines, and the second subproblem grouped machine into cells. Then the subproblems are solved using heuristic graph partitioning procedure. Finally, an approach to determine the economic batch size is also included.

Faber and Carter (1986) developed a graph theoretic algorithm for grouping machines and parts into manufacturing cells by converting the machine similarity matrix into cluster network. The cluster network is partitioned into cells by solving a minimum cost flow problem.

Vohra *et al.* (1990) proposed a network-based algorithm to minimize the amount of machining times performed outside the part primary cells. A non-heuristic network

approach is used to form manufacturing cells with minimum intercellular interactions. The machine-part matrix containing machining times is represented as a network which is subsequently partitioned by using a modified Gomory-Hu algorithm to find a minimum intercellular interaction.

Askin *et al.* (1991) proposed a formulation for machine and part grouping problem, so called a Hamiltonian Path approach. The part-matrix incidence matrix was used to represent the problem. The Jaccard's similarity measure (Carrie 1973) was used to form a distance measure for each machine pair and part pair. The advantages of this method are that it is independent of the starting point and it overcomes the shortcomings of ROC2. Better results were obtained by the proposed formulation compared to those from binary clustering method.

Wu and Salvendy (1993) developed a network (an undirected graph) model to partition the machine-machine graph into cells by considering operation sequences. Two algorithms are used in this model. The first algorithm partitions the network by finding the minimum cut sets in the network so that the resultant interaction between cells is minimal. The second algorithm is a simplified version of the first algorithm by selecting seed nodes in partitioning the network to further reduce the amount of computation. However, the solution from this method is not guaranteed optimal (minimum intercellular movements).

Singh and Mohanty (1991) developed a method for selecting an efficient path in a fuzzy multi-objective network to solve the routing problem in the manufacturing cell. An application of the methodology was also illustrated as the process plan selection problem.

Kandiller (1998) presented a cell formation technique using the hypergraph representation of the manufacturing systems. The proposed method approximates the hypergraph model by graphs so that the cuts are less affected by the approximation. A Gomory-Hu cut tree of the graph approximation then can be obtained. The minimum cuts between all pairs of vertices are calculated easily by means of this tree, and a partition tree is produced. An algorithm is also presented to cut the partition tree. This algorithm is subjected to an experimentation of randomly generated manufacturing situations.

2.1.8. Artificial Intelligence Approach

Elmaraghy and Gu (1988) presented an approach for using domain specific knowledge rules and a prototype feature-based modeling system to automate the process of identifying parts attributes and assigning the parts to most appropriate manufacturing cells. The expert assignment system is based on the geometric features of the parts, characteristics of formed manufacturing cells, parts functional characteristic and attributes, as well as domain specific manufacturing knowledge.

Kaparthi and Suresh (1991) presented a pattern recognition approach utilizing neural network as an alternative of CF solution methods. It is found that this approach could generate codes accurately and promises to be a useful tool for the automatic generation of shape-based classes and codes.

Kaparthi and Suresh (1992) developed a neural network clustering method for the part-machine grouping problem in GT. Among several neural networks, a Carpenter-Grossberg network was selected due to the fact that this clustering method utilizes binary-valued input and it can be trained without supervision. It is shown that this

adaptive leader algorithm has the capability to handle large industry-size data sets due to the computational efficiency.

Chu (1993) proposed a neural network approach in grouping similar parts into families and corresponding machine into cells. The neural network approach in the clustering is based upon a competitive learning paradigm.

Liao and Gen (1993) developed the adaptive resonance theory (ART1) neural models for group technology part family and machine cell formation. An ART1 neural model was first implemented in language C and was tested with examples taken from the literature. The ART1 model was then integrated with a feature-based design system for automatic GT coding and part family forming. It was finally incorporated into a 3-stage procedure for designing cellular manufacturing systems. The evaluation concludes that ART1, when compared with nonlearning algorithms, is best suited for GT applications due to its fast processing speed, fault tolerance and learning abilities, and easy for classifying new parts.

Liao and Lee (1994) implemented the feature-based CAD system and ART1 neural network system for the automated group technology coding and part family forming. After the design process, parts are denoted by binary vectors. Then the ART1 neural model takes these binary vectors as inputs and forms part families according to the similarities of parts in terms of machining features. Each part and part family are then assigned GT codes according to a customized scheme. These systems have been implemented in C++ and C languages.

Burke and Kamal (1995) developed the fuzzy ART neural network to the part family and machine grouping problem in cellular manufacturing. The fuzzy ART is based

on a similarity measure from fuzzy set theory. This method is applied to deal with several shortcomings of the previous ART networks that have been used for classifying and grouping of similar vectors from a machine-part matrix

Enke *et al.* (1998) adapted and implemented the ART1 neural network on a neuro-computer utilizing 256 processors, allowing the neural network to take advantage of its inherent parallelism. Tremendous improvements in the speed of the machine-part matrix optimization result from the parallel implementation. Some comparisons with the previous serial algorithm are also discussed.

Chung and Kusiak (1994) presented an application of a back-propagation neural network for the grouping of parts. The back-propagation neural network is provided with binary images describing geometric part shapes, and it generates part families. To decrease the chance of reaching a local optimum and to speed up the computation process, 3 parameters – bias, momentum, and learning rate – are taken into consideration. The contribution of the analysis is in the design of a neuro-based system to group parts.

Kiang *et al.* (1995) studied the potential of using the self-organizing map (SOM) network as the clustering tool for part family formation in group technology. The SOM network, a variation of neural computing networks, is based on the observation of the operation of the brain. The technique of SOM is presented and how it may be applied as a clustering tool to GT is shown. A computer program for implementing the SOM neural network is also developed.

Lee *et al.* (1997) proposed a machine cell formation method based on the adaptive Hamming net which is a neural network model. The applicability of the method also

presented based on some experiment results along with the method comparison with other cell formation method.

Gwiazda and Knosala (1997) presented an application of Kohonen net in group technology. This net is used in the main task in GT and namely in classification of parts based on their constructional forms. Solids of prismatic and cylindrical shapes are analyzed as classification objects. The methods of description of the constructional form are also analyzed. This method based on partition of solids on small octants and assignation of special codes on them. The solid partition on octants is the same for all solids.

Pilot and Knosala (1998) presented a classification method based on the Kohonen network and its modifications. The application used a 2-layer parallel net which allows joining of both geometrical technological features. The value range is chosen to optimize the net parameters and the quality of classification of machine element input patterns. Input patterns are written in the form a raster grid, in which every raster has a defined number code.

Liang and Zolfaghari (1999) proposed the application of an ortho-synapse Hopfield neural network approach ($OSHN_g$) to solve the comprehensive grouping problems. The comprehensive grouping problems here are the machine cell formation problems with considering processing times and machine capacities. In industrial design problems, which have a large number of parts and machines involved, these problems become a NP-complete which cannot be solved in polynomial time. The results show that the $OSHN_g$ method is very efficient and its solution quality is comparable to that of a simulated annealing approach.

Mahdavi *et al.* (2001) developed a graph-neural network approach to deal with the cell formation problems in group technology. The graph-neural network has demonstrated its advantages of fast computation and its ability to handle large scale industrial problems without the assumption of any parameter and the least exceptional elements in the presence of bottleneck machines and/or bottleneck parts.

2.1.9. Mathematical Programming Approach

Mathematical programming approach in part family and machine cell formation has been developed formulated by using linear programming (LP), linear and quadratic integer programming (LPQ), dynamic programming (DP), and goal programming (GP). There are several objectives in modeling using the mathematical programming approach, such as minimization of total cost function, maximization of the similarity measures while considering different system constraints such as capacity and budget limitations. Furthermore, these models are categorized based on their objectives into two main groups: Models for Cell Formation and Models for Minimizing System Costs.

2.1.9.1. Models for Cell Formation

The models discussed in this section, analyze the part family and machine cell formation problem either sequentially or simultaneously. The objectives of the models include, maximizing the similarity measure within each part family and/or machine cell, minimizing the intercellular movements, and the deviations between the workload assigned to the machine and the available capacity. System constraints, such as available capacity and cell size, are also under consideration.

Kumar *et al.* (1986) developed a 0-1 quadratic programming with linear constraints to solve the problem of grouping parts and components in flexible manufacturing cells. The quadratic model has been converted to two linear problems and dealt with the *k*-decomposition problem. The algorithms developed are suitable for computer implementation and large problem to find an initial solution and for refining this solution.

Kusiak (1987) developed a generalized group technology concept based on generation for one part of a number of different process plans. This concept improves the quality of process (part) families and machine cells. Two classes of CF models are considered: matrix formulation and integer programming formulation (*p*-median model). In the matrix formulation, judgment regarding the number of clusters and the number of elements in each cluster is performed by human, while in the integer programming formulation both of them are determined by clustering algorithm.

Nagi *et al.* (1990) developed a linear programming formulation to solve the manufacturing cell design problem with multiple routings and multiple functionally workcentres. The LP-based model simultaneously addressed two problems: routing selection and cell formation. Minimizing the intercell traffic in the systems was the main objective of the proposed model. And the algorithm iteratively solved two problems: the LP formulation and the Inter-Cell Traffic Minimization Method (ICTMM).

Song and Hitomi (1992) presented a quadratic assignment problem (QAP) formulation to minimize the total number of parts produced in more than one cell (minimize the intercell movements). This problem was solved using both Lagrangean relaxation technique and the optimality conditions of quadratic program. Then the

branch-and-bound algorithm was employed to find the global optimal solution instead of local optimal solution.

Dahel and Smith (1993) proposed two models to solve the cellular manufacturing systems problem. First, a 0-1 integer programming models to minimize the intercell moves subject to machine capacity and cell size constraints. Second, a multiobjectives model is used to form cells which are both flexible and have minimum interactions. Both models group parts and machines simultaneously into the number of cells selected by the cell designer.

Süer (1996) proposed a two-phase hierarchical methodology to find the optimal manpower assignment and cell loads simultaneously. Mixed integer and integer programming formulations are used to generate alternative operator levels and to achieve the optimal operator and product assignment to the cells. The methodology is illustrated with an example problem drawn from a real manufacturing company. The methodology remains valid for most labor-intensive manufacturing cells.

Moon and Gen (1999) proposed an approach for designing independent manufacturing cells with alternative process plans and machine duplication consideration. The problem is formulated as a 0-1 integer programming model and solved using genetic algorithm. It determines the machine cell, part family and process plan for each part simultaneously. Several manufacturing parameters, such as production volume, machine capacity, processing time, number of cells, and cell size, are considered in the process.

Sofianopoulou (1999) formulated a mathematical programming model to deal with alternative process plans and/or machines replication in the cell formation problems. A specially adapted two-dimensional simulated annealing heuristic has been developed

and utilized to get enhanced system configurations of random instances of medium sized production systems.

Won (2000) proposed two *p*-median models using new measures of similarity between machine pairs. The first one with the prespecified number of cells, and the second one without the prespecified number of cells. These models are developed to deal with critical disadvantages of the previous model, such as: too many binary variables and constraints that required by the model, and the number of part families must be known in advance.

Won and Lee (2001) developed two types of production data-based part-machine incidence matrices (PMIMs) to reflect the real-field data such as the operation sequences and production volumes of the parts. The type I production data-based PMIM represents the actual flows to or from machines by parts. A 0-1 linear formulation using the type I production data-based PMIM is developed to minimize the total intercell flows. And the type II production data-based PMIM is constructed to reflect the actual intercell flows by bottleneck parts requiring operations outside its corresponding cell. Computational results show that the proposed production data-based PMIMs can be used as the fundamental replacements over the classical binary PMIM in the cell formation problems.

Baykasoğlu *et al.* (2001) developed an integer multiple objective non-linear mathematical programming formulation for simultaneously forming part / machine cells. In the model, generic capability units which are termed as resource elements are used to define the processing capabilities of machine tools. The model also considers several important objectives, such as minimization of part dissimilarity (based on production requirements and processing sequence) in formed cells, minimization of cell load

imbalance, and minimization of extra capacity requirements for cell formation. A simulated annealing algorithm is developed to solve the model.

2.1.9.2. Models for Minimizing System Costs

The objective of the models discussed in this section is to minimize the system costs in the part families and machine cell formation. These costs usually include annual setup costs, average annual work-in-process costs, annual average production costs, capital machine investment costs, material handling costs, and subcontracting costs. System constraints such as available machining capacity, cell size limits and budget restrictions are also under consideration.

Purcheck (1985) proposed a linear-programming-based heuristic method to solve the cell formation problem. The research focused on the planning and study of machine-component group in flexible production cells and flexible manufacturing systems. The heuristic is designed to search the solution space of the problem in monotone-increasing order of solution costs so as to avoid the enumeration of solutions for cost minimization.

Choobineh (1988) proposed a two-stage procedure for the design of a cellular manufacturing system. The first stage forms the part families. And the second stage forms the machine cells. An integer programming model is developed for this stage. The objective function includes the total of annual setup costs, average annual WIP costs, annual average production costs, and machine investment costs. The relevance of this approach in the design of flexible manufacturing systems was also discussed. However, the model only considers the available machining capacity and available operating budget restrictions.

Rajamani *et al.* (1990) developed three integer programming models to successively study the effect of alternative process plans and simultaneous formation of part families and machine groups. The models are presented to analyze how alternative process plans influence the resources utilization when the part families and machine groups are formed simultaneously. The first model assigns machines to parts, with the objective function to minimize the total investment cost under the machine capacity and budget constraint. The second model selects the process plan for each part, machine type for each operation, and number of machines of each type in different cells. The objective function of the second model is to minimize the total investment on machines of different types assigned to the cells. The third model identifies part families and machine groups simultaneously. Comparisons of the three models based on cost functions are also given.

Askin and Chiu (1990) proposed a cost-based mathematical model for the grouping of individual machines into cells and the routing of components to machines within cells. The mathematical programming formulation involved cost of inventory, machine depreciation, machine setup, and material handling costs. A two stage formulation is developed to simplify the solution procedure. The first stage assigns the operations of parts to the individual machines, and the second stage assigns the machines to the groups, with the objective of minimizing intercellular material handling cost.

Jain *et al.* (1990) discussed a 0-1 integer programming model to solve the cell formation problem in flexible manufacturing systems under resource constraints. The model was developed to form the machine-part groups and to decide on the number of machines and the number of copies of tools required to achieve minimum overall system

cost. The model takes into account the processing time available on any machine, tool lives and the processing requirements of the parts.

Alfa *et al.* (1992) developed a mathematical model for simultaneous solution of the machine grouping and layout problems in cellular manufacturing systems. Since the model is complex to solve using traditional optimization techniques, a simulated annealing algorithm is used to find suboptimal solution. An alternative formulation based on quadratic assignment problem was also presented.

Shafer *et al.* (1992) presented a mathematical programming model that deals with exceptional elements, which are bottleneck machines and exceptional parts that span two or more manufacturing cells. Three important costs were considered in the model: intercellular transfer, machine duplication, and subcontracting. Then, the mathematical programming model is solved to determine how best to deal with the exceptional elements.

Tsai *et al.* (1997) developed a fuzzy mixed-integer programming model (FMIP) to simultaneously form manufacturing cells and minimize the cost of dealing with exceptional elements. The authors also illustrated how a FMIP approach can be used to solve the cell formation problem in a fuzzy environment, proposed a fuzzy operator, and examined the impact of different membership functions and operators on computational performance.

Sarker and Li (1997) presented a mixed-integer programming model to simultaneously select part routings and form machine cells in the presence of alternate process plans so that the total cost of operating and intercell material handling is

minimized. Demand for parts, machine capacities, number of cells to be formed, and number of machines in a cell are included in the model.

Zhou and Askin (1998) studied the formation of general group technology cells based on the operation requirements of parts and operation capabilities of machines. Parts are first grouped into families by using a similarity coefficient based on common operation types. An integer model is then developed to solve the problem of machine group selection. The models include machine cost, variable production cost, setup cost, and intracell material handling cost. A minimum increment heuristic and a simulated annealing heuristic are proposed for solving the model more efficiently.

Kamrani *et al.* (1998) presented a simulation-based methodology which uses both design and manufacturing attributes to form manufacturing cells. The methodology is implemented in three phases. In phase I, parts are grouped into part families based on their design and manufacturing dissimilarities. In phase II, machines are grouped into manufacturing cells based on relevant operational costs and various cells are assigned part families using an optimization technique. Phases I and II are based on integer and mixed-integer mathematical models. Finally, in phase III, a simulation model of the proposed system is built and verified, and the model is run so that data on the proposed system may be gathered and evaluated.

Salum (2000) proposed a two-phase method based on total manufacturing lead time (MLT) reduction to minimize the cost. In the first phase, the system is simulated by considering all of the operational issues under the assumption of zero material handling times to minimize total MLT. Besides minimized total MLT, the first phase also yields the waiting times of parts and the volume of the parts-flow between machines, which are

used to find similarity measures between machines in the second phase. The second phase then exploits an algorithm which creates and uses these similarity measures to construct a layout by locating machines with higher similarity next to each other to justify the assumption of minimal total MLT and to minimize total material handling time.

Mak and Wong (2000) developed a mathematical model to maximize the intracell and minimize the intercell part movements in the manufacturing system, and to ensure the balance of part movements within each machine-part cell. The mathematical model is then solved using the genetic algorithms approach. The effectiveness of the approach is demonstrated by using and industrial example.

Adenzo-Díaz *et al.* (2001) developed a mathematical model with the minimization of transportation costs as the objective function. Limit on cell sizes as well as separation constraint and co-location constraint may be imposed. A Tabu Search algorithm is used to solve this problem. Extensive computational experiences with large-size problems show that this method outperforms some existing Simulated Annealing approaches.

2.1.10. Human Issues in Cellular Manufacturing Systems

Previous research publications indicated that human or worker issues have been considered into the application of Cellular Manufacturing Systems. The literature review regarding this matter is presented in this section.

Huber and Brown (1991) stated that most research on cellular manufacturing has focused on the technical issues. Based on sociotechnical systems (STS) theories, it is

suggested that the development of cellular manufacturing social system be pursued with the same priority as the changes in the technical systems.

Min and Shin (1993) developed a mixed-integer goal programming model to deal with the human cells in group technology. The model considered that the cell formation problem usually entails various labor-related problems, so that to gain full benefit of group technology both machine and human cells should be formed simultaneously. The model recognized multiple conflicting objectives to perform a comprehensive cost-benefit analysis of group technology application. The human issues considered here mostly concentrated on the worker technical skill and capability. There is no ergonomics concern included in this paper.

Maffei and Meredith (1994) performed an in-depth study of six companies using flexible manufacturing cells. In this study, attention to human issues or workers within cells are still important, even though new popular technology (such as JIT) have been reported to reduce the workers autonomy. Furthermore, it was also concluded in the guidelines that the more operator involved and responsible in the systems, the smoother the operations run.

Chen (1995) examined the operator scheduling problems in cellular manufacturing systems. The role of operator is to load jobs into machines, to unload jobs from machines, and to transport jobs between machines. Since the productivity of operator is relevant with the productivity the system, the operator scheduling also become a critical issue. However, this paper only discussed the human issues in the area of operator scheduling.

Fan and Gassmann (1995) conducted a series of pilot studies in a medium sized manufacturing company which makes braking systems for heavy goods vehicle. The study concluded that the social problems involved with sustaining the performance of a cell, meeting the needs of individual people, accommodating individual differences, and trying to improve the working environment are enormous. It was also concluded that the new form of work organization and information technology much be developed through multi-disciplined research projects to overcome the long-term impact of human centered concepts.

King and Majchrzak (1996) studied the used of Concurrent Engineering (CE) tools to increase the concurrency of multidisciplinary design by integrating various technologies (such as computer-aided-design, computer-aided-design, expert system, and communication networks), and their assumptions regarding the human factors. This study concluded that the assumption given in the CE tools has ignored the human issues. Furthermore, CE tool developers should follow a user-centered system development process and design tools that incorporate human factors component.

Askin and Huang (1997) develop two integer programming models for guiding the assignment of workers to cells and the selection of a training program for each employee. The first model requires less data and provides an aggregate training program for each cell and a worker assignment plan, and this model also ensures that an adequate supply of skilled workers is assigned to cell. The second model uses more detailed information and gives a specific training plan and working time dedicated to every skill for each worker as well as a worker assignment plan.

Warner *et al.* (1997) presented a research plan to develop a model to assign workers to manufacturing cells based on their technological and human issues skill. The model addresses human issues such as hiring and firing, absenteeism, training, and compensation.

Eckstein and Rohleder (1998) presented a simulation study which incorporates the human resources issues in group technology in a comprehensive setting. The study based on the previous research that ignored some of the most basic operating conditions like human resources issues, such as learning and labor constraints. The study discovered that group technology is significantly better than a job shop when considering human resource issues.

Sohal *et al.* (2001) presented a case study based on the application of cellular manufacturing in a medium-sized Australian manufacturer. The case study revealed that the effective management of human resources in the flexible cellular environment is required to ensure strategic benefit over a long period of time. It was also concluded that the human element of cellular manufacturing is equally important.

Olorunniwo and Udo (2002) used the sociotechnical system (STS) principles to identify major factors that are likely to impact the cellular manufacturing implementation. Those factors are top management role, job design for operators, and cross training. One of the conclusions of the study is that the human resources need to be planned to get a good job satisfaction, morale level, and quality of work life.

Norman *et al.* (2002) formulated a mixed integer programming model to solve the problem of assigning workers to manufacturing cells that considers both human and technical skills and their impact on system performance. The system performance is

defined as the maximization of profit where profit comprises productivity, quality costs, and training costs associated with a particular worker assignment. The worker assignments here are not only based on the technical skill of the worker, but also include the human skills and permits the ability to change the skill level by providing additional training.

2.2. Human Limitations on Manual-Material-Handling Activities

Human limitations on manual-material-handling activities based on the Revised NIOSH Lifting Guidelines are presented in this section. The NIOSH Lifting Guidelines can be considered as one of the most widely used lifting guidelines to deal with human limitations on manual lifting tasks. Presented below is the brief overview about the history of the Revised NIOSH Lifting Index taken from Waters and Putz-Anderson (1998).

Historically, the *National Institute for Occupational Safety and Health* (*NIOSH*) has recognized and addressed the problem of work-related back injuries and published the *Work Practices Guide for Manual Lifting* (*WPG*) in 1981 (refer as NIOSH 1981). The *WPG* contains a summary of the lifting-related literature up to 1981, analytical procedures, a lifting equation for calculating a recommended weight for specified two-handed, symmetrical lifting tasks, and an approach for controlling the hazards of low-back injury from manual lifting. The approach to hazard control was coupled with the action limit (AL), a term that denoted the recommended weight derived from the lifting equation.

In 1985, NIOSH convened an ad hoc committee of experts who reviewed the current literature on lifting, including the NIOSH *WPG*. The literature review was summarized in a document containing updated information on the physiological, biomechanical, psychophysical, and epidemiological aspects of manual lifting (NIOSH 1991). Based on the results of the literature review, the ad hoc committee recommended criteria for defining the lifting capacity of healthy workers. The committee used the criteria to formulate the revised lifting equations. Subsequently, NIOSH staff developed documentation for the equation and played a prominent role in recommending methods for interpreting the results of the lifting equation. The revised lifting equation reflects new findings and provides methods for evaluating asymmetrical lifting tasks and lifts of objects with less than optimal couplings between the object and the worker's hands. The revised lifting equation also provides guidelines for a more diverse range of lifting tasks than the earlier equation (NIOSH 1981).

Although the revised lifting equation has not been fully validated, the recommended weight limits derived from the revised equation are consistent with, or lower than, those generally reported in the literature. Moreover, the proper application of the revised equation is more likely to protect healthy workers for a wider variety of lifting tasks than methods that rely solely on a single task factor or criterion.

The section 2.2.1 below presented the literature review on the application and evaluation of the NIOSH lifting guidelines. And the section 2.2.2 and 2.2.3 presented the summary of the revised NIOSH lifting equation. The first sub-section discusses about the lifting index for single-task lifting. And the second sub-section presents the composite lifting index for multi-task lifting.

2.2.1. Application and evaluation of NIOSH Lifting Guidelines

Wang *et al.* (1998) performed an epidemiological study to evaluate the relationship between low-back discomfort ratings and use of the revised NIOSH lifting guide to assess the risk of manual-material-handling (MMH) tasks. 97 MMH workers were surveyed on site in 15 factories and designed a questionnaire to systematically collect job-related information. Approximately 90% of the workers had suffered various degrees of lower back discomfort, and 80% had sought medical treatment. The survey showed that 42 of 97 jobs analyzed had a recommended weight limit of 0, which was attributed to either a horizontal distance or a lifting frequency that exceeded the bounds of the NIOSH lifting index. For the remaining 55 jobs, the significant positive correlation obtained between the lifting index and the severity of low-back discomfort suggests that the lifting index is reliable in accessing the potential risk of low-back injury in MMH. These findings provide useful information on the application of the NIOSH lifting guide to the assessment of low back pain.

Waters *et al.* (1998) studied the accuracy of measurements for the revised NIOSH lifting equation. 27 non-ergonomists who participated in a one-day training session on the use of NIOSH lifting equation were subsequently tested on a simulated lifting task eight weeks later to determine their accuracy in measuring variables. In the summary, the study has demonstrated that it is possible to train individuals in one day to accurately make the measurements required to properly use the revised NIOSH lifting equation to determine recommended weight limit and lifting index values for certain two-handed manual lifting tasks.

Chung and Kee (2000) presented an application of the NIOSH lifting equation on a fire brick manufacturing company with a high prevalence of low back injuries. Several manufacturing processes were analyzed: forming, heating, and packing processes involving frequent lifting and lowering in asymmetric postures. Composite Lifting Indices (CLIs) based on the revised NIOSH lifting equations were calculated for 14 tasks of the forming process and 5 tasks of the heating/packaging processes. Calculated CLIs for the tasks ranged from 0.86 to 8.8 (average 2.73) in forming process and from 3.7 to 18.9 (average 11.12) in the heating/packaging processes. The majority of the lifting tasks in the company exceeded the recommended weight limit (RWL). The results suggest that the tasks should be redesigned ergonomically to eliminate the risk factors that may cost low back injuries.

Lee *et al.* (1996) investigated the application of the revised NIOSH equation for Korean workers. Using the psychophysical methodology, young male college students and field workers were examined in the experiment. The psychophysical experiment and the validation experiment were performed in sagittal plane where lifting frequency and lifting height varied. The load constant obtained in this investigation was about the same as the one recommended in the NIOSH equation, which means that young and healthy Korean males are well protected by the revised NIOSH equation.

2.2.2. Revised NIOSH Lifting Guidelines

The revised NIOSH lifting guidelines consist of the basic lifting index formula, the recommended weight limit, and all the multipliers. These guidelines were adopted from Konz and Johnson (2000).

2.2.2.1. Basic Formula

The *lifting index* (*LI*) offers a relative estimate of the level of physical stress associated with a particular manual-lifting task. The estimate of the level of physical stress is defined by the relationship between the weight of the load lifted and the recommended weight limit. The *LI* is defined by the equation:

$$LI = LW / RWL \tag{2.1}$$

LI = Lifting index, proportion

Ideal value $LI < 1.0$

LW = Load / weight of item lifted, lb or kg

RWL = Recommended weight limit, lb or kg

2.2.2.2. Recommended Weight Limit

The principal product of the revised NIOSH lifting equation is the *recommended weight limit* (*RWL*). The *RWL* is defined for a specific set of task conditions as the weight of the load that nearly all healthy workers could perform over a substantial period of time (up to eight hours) without an increased risk of developing lifting-related low-back pain (LBP). "Healthy workers" are those who are free of adverse health conditions that would increase their risk of musculoskeletal injury.

The concept behind the revised NIOSH lifting equation is to start with a recommended weight that is considered safe for an "ideal" lift (load constant equal to 51 lb or 23 kg) and then reduce the weight as the task become more stressful (the task-related factors become less favorable). The precise formulation of the revised lifting

equation for calculating the *RWL* is based on a multiplicative model that provides a weighting (multiplier) for each of six task variables:

- horizontal distance of the load from the worker (*H*),
- vertical height of the lift (*V*),
- vertical displacement during the lift (*D*),
- angle of asymmetry (*A*),
- frequency (*F*) and duration of lifting, and
- quality of the hand-to-object coupling (*C*).

The *RWL* is defined as:

$$\boldsymbol{RWL = LC \times HM \times VM \times DM \times FM \times AM \times CM} \tag{2.2}$$

LC = Load constant = 51 lbs or 23 kg

HM = Horizontal multiplier, proportion

VM = Vertical multiplier, proportion

DM = Distance multiplier, proportion

FM = Frequency multiplier, proportion

AM = Asymmetry multiplier, proportion

CM = Coupling multiplier, proportion

Horizontal Multiplier

$$HM = BIL / H \tag{2.3}$$

BIL = Body interference limit = 10" (25 cm)

H = Horizontal distance from the large knuckle at the end of the third finger to the ankle midpoint (midpoint of inner ankle bones)

The maximum value of H (functional reach limit) is 25" (63 cm)

Vertical Multiplier

$$VM = 1 - VC \mid V - KH \mid \tag{2.4}$$

VC = Vertical constant = 0.0075" (0.003 cm)

V = Initial vertical height of knuckles when lifting object

The maximum value of V is 70" (175 cm)

KH = Knuckle height of typical lifter (assumed stature height of 66" (165 cm))

= 30" (75 cm)

When $V \leq 30$", the lifting is considered as whole body

when $V > 30$", the lifting is considered as upper body

The concept is a 22.5% penalty for lifts from the floor or shoulder (60" or 150 cm).

Distance Multiplier

$$DM = .82 + DC / D \tag{2.5}$$

.82 = Multiplier at maximum hand height of 70" (175 cm)

DC = Distance constant = 1.8" (4.5 cm)

D = Distance moved vertically (absolute value) or vertical travel distance

If $D \leq 10$" (25 cm), the value of $DM = 1.0$

Maximum $D = 70 - V$ (for inches) or $D = 175 - V$ (for cm)

Frequency Multiplier

The Frequency Multiplier for NIOSH lifting equation is shown in Table 2.1. Lifting frequency can range from less than 1 in 5 minutes (0.2 lifts/min) to 15 lifts/min. It is the mean number of lifts in a 15-minute period. If lifting is not continuous, count the number of lifts in 15 minutes and divide by 15. The frequency multiplier varies depending on lifting duration/session (D) and whether the initial vertical location of the hands (V) is above or below typical knuckle height (30" or 75 cm).

Lifting duration/session in hours has three categories:

- Short = 0.001 h $\leq D \leq$ 1 h, with recovery time of at least 1.2D,
- Moderate = 1 h $< D \leq$ 2 h, with recovery time of at least 0.3D,
- Long = 2 h $< D \leq$ 8 h.

During recovery time, the person is resting or has light work (such as sitting, standing, walking, monitoring). If a person does not meet the recovery criterion, omit the recovery time and add the work times together.

This Frequency Multiplier is adapted into a linear equation with assumption that the lifting is performed in a long lifting duration and lower body lifting ($V < 30$"). The linear regression result is shown on Figure 2.1.

Table 2.1: Frequency multiplier (Konz and Johnson 2000)

Frequency (lift/min)	$D \leq 1$ hour		1 hour < $D \leq 2$ hours		2 hours < $D \leq 8$ hours	
	$V < 30$"	$V \geq 30$"	$V < 30$"	$V \geq 30$"	$V < 30$"	$V \geq 30$"
≤ 0.2	1.00	1.00	.95	.95	.85	.85
0.5	.97	.97	.92	.92	.81	.81
1	.94	.94	.88	.88	.75	.75
2	.91	.91	.84	.84	.65	.65
3	.88	.88	.79	.79	.55	.55
4	.84	.84	.72	.72	.45	.45
5	.80	.80	.60	.60	.35	.35
6	.75	.75	.50	.50	.27	.27
7	.70	.70	.42	.42	.22	.22
8	.60	.60	.35	.35	.18	.18
9	.52	.52	.30	.30	.00	.15
10	.45	.45	.26	.26	.00	.13
11	.41	.41	.00	.23	.00	.00
12	.37	.37	.00	.21	.00	.00
13	.00	.34	.00	.00	.00	.00
14	.00	.31	.00	.00	.00	.00
15	.00	.28	.00	.00	.00	.00
> 15	.00	.00	.00	.00	.00	.00

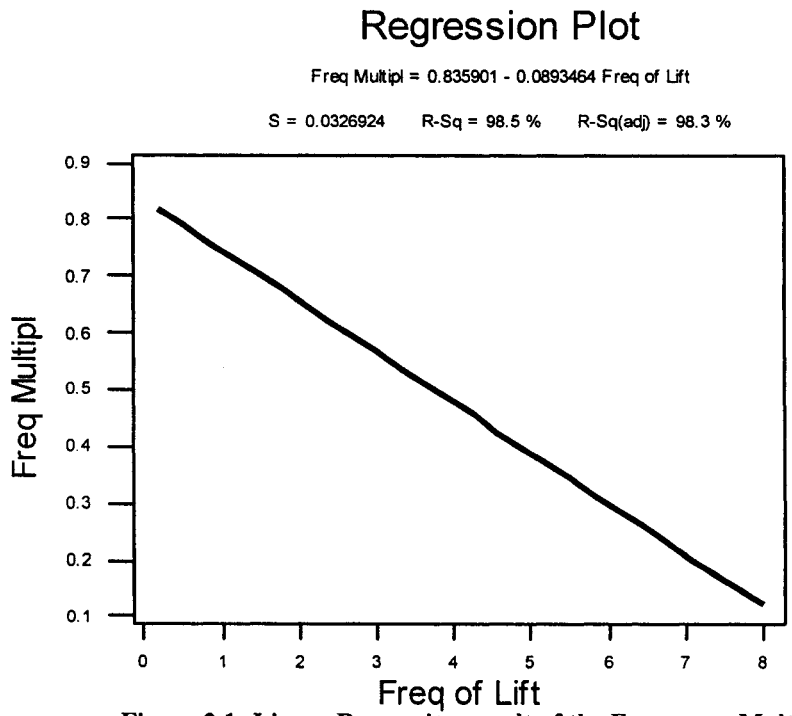

Figure 2.1: Linear Regression result of the Frequency Multiplier

From the linear regression on Figure 2.1, the formula for the Frequency Multiplier:

$$FM = 0.836 - 0.089\,F \tag{2.6}$$

F = Frequency of Lifting (lifts per minute)

Asymmetry Multiplier

$$AM = 1 - 0.0032\,A \tag{2.7}$$

A = Angular deviation / degrees (angle of symmetry) of the midpoint of the two hands from straight ahead (neutral body posture; sagittal plane)

A can range from 0 to 135° (ignore clockwise or counter-clockwise). The concept is a 30% penalty for a 90° angle.

Coupling Multiplier

Table 2.2 provides the Coupling Multiplier and Figure 2.2 shows the decision tree for hand/object coupling condition . This multiplier depends on the height of the initial and final hand-container coupling and whether the coupling is good, fair, or poor.

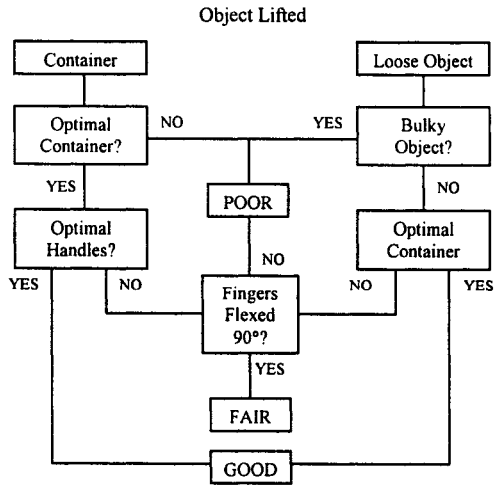

Figure 2.2: Decision tree for hand/object coupling condition (Konz and Johnson 2000)

Table 2.2: Coupling multiplier (Konz and Johnson 2000)

Couplings	$V < 30"$	$V \geq 30"$
Good	1.00	1.00
Fair	0.95	1.00
Poor	0.90	0.90

Table 2.3: Hand-to-Container Coupling classification and definitions of optimal container, handle, and cut-out (Konz and Johnson 2000)

Good	Optimal container design with optimal handles or optimal handhold cut-outs Loose parts or irregular objects with comfortable grip (hand can easily be wrapped around object)
Fair	Optimal container design with non-optimal handles or non-optimal handhold cut-outs Loose parts with no handles or handhold cut-outs
Poor	Non-optimal design containers with no handles or handhold cut-outs Loose parts or irregular objects that are bulky or hard to handle

	Design	Optimal	Non-Optimal
CONTAINER	Frontal length	≤ 16" (40 cm)	Non-optimal is failure to meet one or more optimal condition
	Height	≤ 12" (30 cm)	
	Surface	Smooth, non-slip	
	Edge	Non-sharp	
	Center of mass	Symmetric	
	Load	Stable	
	Gloves required	No	
HANDLE	Diameter	0.75" to 1.5" (1.9 to 3.8 cm)	Non-optimal is failure to meet one or more optimal condition
	Length	≥ 4.5" (11.5 cm)	
	Clearance	≥ 2" (5 cm)	
	Shape	Cylindrical	
	Surface	Smooth, non-slip	
HANDHOLD CUT-OUT	Height	≥ 3.8" (1.5 cm)	Non-optimal is failure to meet one or more optimal condition
	Length	≥ 4.5" (11.5 cm)	
	Clearance	≥ 5" (2 cm)	
	Container thickness	≥ 0.43" (1.1 cm)	
	Shape	Semi-oval	
	Surface	Smooth, non-slip	

2.2.2.3. Formula for Multi-Task Lifting

Multi-task lifting consists of several single-tasks lifting with different lifting condition and/or different weight of lifting.

The multi-task procedure is (for i number of task):

1. Compute two version of the recommended weight limit for each task:

   Frequency-Independent RWL (FIRWL): RWL but set the frequency multiplier to 1.

   $$FIRWL_i = RWL_i / FM_i \tag{2.8}$$

   Single-Task RWL (STRWL): RWL for single task.

   $$STRWL_i = RWL_i \tag{2.9}$$

2. Compute two version of lifting index for each task:

   Frequency-Independent Lifting Index (FILI):

   $$FILI_i = LW_i / FIRWL_i \tag{2.10}$$

   $LW_i =$ Load / weight of task i

   If the load varies, use the average of the various loads.

   Single-Task Lifting Index (STLI):

   $$STLI_i = LW_i / STRWL_i \tag{2.11}$$

3. Composite Lifting Index (CLI):

   Rank the tasks in order of the STLI value (the greatest value first / descending sort). Give the index r to the ranked tasks (start from the greatest, so $r = 1$ is the task index with the greatest STLI). Then compute the CLI with this formula:

$$CLI = STLI_1 + \sum_{r=2}^{i} FILI_r \left[\frac{1}{FM\left(\sum_{s=1}^{r} F_s\right)} - \frac{1}{FM\left(\sum_{t=1}^{r-1} F_t\right)} \right] \tag{2.12}$$

$STLI_1$ = The greatest Single-Task Lifting Index (rank #1)

$FILI_r$ = Frequency-Independent Lifting Index for task rank #r

$FM(F)$ = Frequency Multiplier for F lifting frequency

2.2.3. Applying the equations

The recommended weight limit (*RWL*) and lifting index (*LI*) can be used to guide ergonomic design in several ways (Waters and Putz-Anderson 1998):

1. The individual multipliers can be used to identify specific job-related problems. The relative magnitude of each multiplier indicates the relative contribution of each task factor (horizontal, vertical, frequency).
2. The *RWL* can be used to guide the redesign of existing manual lifting jobs or to design new manual lifting jobs. For example, if the task variables are fixed, then the maximum weight of the load could be selected so the *RWL* is not exceeded; if the weight is fixed, then the task variables could be optimized so that the weight is not exceeded by the *RWL*.
3. The *LI* can be used to estimate the relative magnitude of physical stress for a task or job. The greater the *LI*, the smaller the fraction of workers capable of safely sustaining the level of activity. Thus, two or more job designs could be compared.
4. The *LI* can be used to prioritize ergonomic redesign. For example, a series of suspected hazardous jobs could be rank-ordered according to the *LI*, and a control strategy could be developed (jobs with lifting indices above 1.0 or higher would benefit the most from redesign).

Chapter 3 – Model Development

This chapter presents several mixed integer mathematical programming models for part family, machine cell formation, and operator assignment problems in cellular manufacturing systems while considering the human limitation of manual lifting. The operators are assigned to the machine cells to operate the machines and perform the loading and unloading of the parts.

The first group of the models consists of two cell formation with NIOSH lifting index models. The first model in this group evaluates trade-offs among machine duplication, number of operators, machine idle time, operator idle time, and value of composite lifting index under single period planning horizon. The second model in this group is an extension to the first model, which incorporates the part operation sequence, therefore the intercellular movements are allowed. The second group of the models is developed to solve problems under multi period planning horizon.

Accordingly, this chapter is divided into two main sections, one for each model group. In these sections, the general definition of the problems, statement of assumptions, explanation of the variables, parameters used in the design, and development of the models are included.

3.1. Mathematical Models under Single Period Planning Horizon

3.1.1. Problem Definition

The mathematical models developed in this section attempt to form part-family and machine-grouping with operator assignment that does not exceed a certain limit of frequency of lifting and composite lifting index. The objective function is to minimize the total cost that consists of machine procurement cost, operator cost, machine idle time cost, operator idle time cost, intercellular movements cost (for Model 2) and lifting risk cost. Lifting risk cost here can be defined as the expected injury cost based on the composite lifting index. Higher lifting index means higher risk for the operators to be injured. Injury operators can cause additional cost because of several factors such as injury compensations, training cost for new operators, or even maybe the company has to face a lawsuit. Lower lifting index gives a lower injury risk for the operators but inefficient in terms of operator work rate. For example, assigning a high number of operators for a simple lifting task can get a very low lifting index so that low lifting risk cost can be achieved, but an increase in the assigned number of operators also increases the operator cost.

3.1.2. Assumptions

1. Parts to be produced for the period are known and have constant demand throughout the period.
2. For each part, type of machines required and processing times on these machines are available.
3. For Model 1, each part is assigned only to one cell (no intercellular movements).

For Model 2, each part is assigned based on the operation sequence (intercellular movements allowed).

4. For each part, the weights of lift for every operation are the same (constant weight).
5. Machines are in 100% capacity and various costs are known.
6. There can be multiple units of machines from each type in the system and the cost of each machine is assumed to be the same for a given machine type.
7. Number of machine cells to be formed is determined before.
8. All machine cells are manual labor intensive (need operators to operate machines and to work on manual material handling).
9. Operator cost is the same for every operator assigned and operators are in 100% availability.
10. Expected injury cost per operator is known and calculated proportionally to the composite lifting index of the cell where the operator is assigned.
11. Lifting control is needed both in initial condition and final condition.
12. Frequency of lifting for one operator cannot exceed 3 lifts/min.
13. The Frequency Multiplier is adapted from Table 2.1 to a linear equation shown in Figure 2.1.
14. Lifting horizontal distance is below 25 cm, so that the Horizontal Multiplier is 1.
15. The highest vertical lifting distance from knuckle is 50 cm.
16. Vertical lifting distances for each part are known.
17. Lifting position is in the sagittal plane / neutral body posture / symmetrical condition, so that the Asymmetry Multiplier is 1.

18. The object is lifted in optimal container design with optimal handles that present a good coupling, so that the Coupling Multiplier is 1.
19. The composite lifting index for multi-task lifting is adapted from the rank algorithm to a formula.
20. The composite lifting index critical value is 1.5, which means the composite lifting index for each cell cannot exceed 1.5.

3.1.3. Model 1: Single Period

3.1.3.1. Nomenclature

Indices:

p = parts index ($p = 1, 2, \ldots\ldots, n_p$)

m = machine types ($m = 1, 2, \ldots\ldots, n_m$)

c = cells index ($c = 1, 2, \ldots\ldots, n_c$)

Parameters:

D_p = average demand for part p per period, in units

F_m = amortized cost per period to procure one machine of type m

J_m = idle time cost per unit time of machine type m

t_{pm} = unit processing time of part p on machine m

G_m = proportion of operator attention required while one machine type m is operating

LW_p = load weight of part p, in kgs

LD_p = vertical lifting distance of part p, in cm = $| VK_p \text{ final} - VK_p \text{ initial} |$

U = total shop time per period, in hours

S_c = maximum number of machines allowed in cell c

VW_c = maximum number of operators allowed in cell c

N = cost of one operator

JW = idle time cost of operator per unit time

LC = expected operator injury cost per operator per unit Composite Lifting Index

VK_{pc} = vertical distance of part p from knuckle in cell c (choose greater VK between initial and final condition), assumed to be 50 cm

L_{pm} = $\begin{cases} 1 & \text{if } t_{pm} > 0 \text{ (there is operation performed in this machine)} \\ 0 & \text{if } t_{pm} = 0 \text{ (there is no operation performed)} \end{cases}$

Decision variables:

0/1 integer variables:

X_{pc} = $\begin{cases} 1 & \text{if part } p \text{ is assigned to cell } c \\ 0 & \text{otherwise} \end{cases}$

General integer variables:

M_{mc} = number of machine type m assigned to cell c

W_c = number of operators assigned to cell c

General variables:

I_{mc} = idle time of machine type m assigned to cell c

WI_c = idle time of operator in cell c

FM_{pc} = frequency multiplier for the lifting frequency of part p in cell c

$STLI_{pc}$ = single-task lifting index of part p in cell c

CLI_c = Composite Lifting Index of cell c

3.1.3.2. Mathematical Model

Minimize cost:

$$Z = \sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(F_m M_{mc} + J_m I_{mc}\right) + \sum_{c=1}^{n_c}\left(N \cdot U \cdot W_c + JW \cdot WI_c\right) + LC\sum_{c=1}^{n_c} W_c CLI_c \qquad (3.1)$$

Subject to:

$$\sum_{c=1}^{n_c} X_{pc} = 1 \qquad \forall\, p \qquad (3.2)$$

$$I_{mc} + \sum_{p=1}^{n_p} D_p t_{pm} X_{pc} = M_{mc} U \qquad \forall\, m \text{ and } c \qquad (3.3)$$

$$\sum_{m=1}^{n_m} M_{mc} \le S_c \qquad \forall\, c \qquad (3.4)$$

$$WI_c + \sum_{m=1}^{n_m} G_m \sum_{p=1}^{n_p} D_p t_{pm} X_{pc} = W_c U \qquad \forall\, c \qquad (3.5)$$

$$W_c \le VW_c \qquad \forall\, c \qquad (3.6)$$

$$\sum_{p=1}^{n_p}\left(\frac{X_{pc} \cdot D_p \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_c}\right) \le 3 \qquad \forall\, c \qquad (3.7)$$

$$CLI_c \le 1.5 \qquad \forall\, c \qquad (3.8)$$

$$X_{pc} \in \{0,1\} \qquad (3.9)$$

$$M_{mc}, W_c \in \text{integer} \tag{3.10}$$

$$M_{mc}, W_c, I_{mc}, WI_c \geq 0 \tag{3.11}$$

Frequency Multiplier:

The Frequency Multiplier was adapted using linear regression with assumption that lifting is performed in a long duration. The detail of the linear regression is in Figure 2.1.

$$FM_{pc} = 0.836 - 0.089\left(\frac{X_{pc} \cdot D_p \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_c}\right) \tag{3.12}$$

Single-Task Lifting Index:

The Single-Task Lifting Index (*STLI*) is calculated based on the assumption that the horizontal multiplier, asymmetry multiplier, and coupling multiplier are equal to 1. The detail explanation of the *STLI* formula is in Chapter 2 (equation 2.11). Vertical multiplier, distance multiplier, and frequency multiplier are corresponding to equation 2.4, 2.5, and 2.6.

Equation 2.11: $$STLI = \frac{LW}{LC \times HM \times VM \times DM \times AM \times FM \times CM}$$

$$STLI = \frac{LW}{LC \times VM \times DM \times FM} \tag{3.13}$$

Equation 2.4: $$VM = 1 - 0.003|V - KH| = 1 - 0.003VK$$

Equation 2.5: $$DM = 0.82 + {}^{4.5}/_{LD}$$

Equation 2.6: $$FM = 0.836 - 0.089F$$

$$STLI_{pc} = \frac{LW_p \cdot X_{pc} \cdot L_{pm}}{23 \times \left[1 - 0.003VK_{pc}\right] \times \left[0.82 + {}^{4.5}/_{LD_p}\right] \times \left[0.836 - 0.089\left(\frac{X_{pc} \cdot D_p \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_c}\right)\right]} \tag{3.14}$$

Composite Lifting Index (CLI):

The Composite Lifting Index formula below is an approximation from the original NIOSH formula (equation 2.12).

$$CLI_c = \max_p STLI_{pc} + 0.25 \times \left[\frac{\left(\sum_{p=1}^{n_p} STLI_{pc}\right) - \max_p STLI_{pc}}{\left(\sum_{p=1}^{n_p} X_{pc}\right) - 1}\right] \tag{3.15}$$

Objective function

The objective of the model is to minimize the various cost functions related to the design of cellular manufacturing systems using the NIOSH lifting index as the ergonomics measure for manual lifting tasks. There are five components considered in the objective function, which are: total machine procurement cost, total machine idle time cost, total operator salary cost, total operator idle time cost, and total lifting risk cost (total expected injury cost of manual lifting).

The first component in the objective function is the total machine procurement cost related with amortized cost per period to procure different types of machines in the

system. The total machine procurement cost can be calculated by the summation of machine procurement cost multiplied by the number of machine for every machine type in every cell.

The second component is the total machine idle time cost which assigns a penalty cost on the unutilized capacity of the machine. The penalty for the idle time of the machine is defined as the opportunity cost of not processing the part on the machine which is equal to the loss of interest on the investment (for example, machine investment, space investment, and maintenance cost). Additionally, assigning penalty cost on the unutilized capacity of the machine will balance the utilization of the machines in the system without creating excess idle capacity. The total machine idle time cost can be calculated by the summation of machine idle time cost per unit time multiplied by the machine idle hours for every machine type in every cell.

The third component is the total operator cost related with operator assignment in the system. The total operator cost can be calculated by multiplying the cost of one operator per unit time by the operator's time per period, for all operators assigned to the system.

The fourth component is the total operator idle time cost which also assigns a penalty cost on the unutilized capacity of the operator. The penalty for the idle time of the operator is defined as the opportunity cost of not operating the machine or manual material handling which is equal to the loss of interest on the investment. The total operator idle time cost can be calculated by the summation of operator idle time cost per unit time multiplied by the operator idle time in every cell.

The fifth component is the total lifting risk cost (total expected injury cost) associated with manual lifting tasks. It is assumed that this cost is correlated with NIOSH lifting index. This cost can be calculated based on the expected injury cost rate per unit lifting index per operator times the lifting task index (NIOSH score) and the number of operators. This cost, as mentioned before, may include injury compensations, training costs for new operators to replace the injured operators, and productivity loss.

Constraints

The model developed in this section considers ten different operating constraint sets related to the cell formation design that has manual lifting tasks.

It was assumed that multiple units of the same machine can be assigned to one cell and each part can be allocated only to one cell. Constraint (3.2) ensures this integrality by allocating each part type p in only one cell c. Constraint (3.3) ensures that each cell has an adequate number of machines of each type to perform its assigned workload. For each type of machine in the cell, the equation consists of the demand for capacity, unutilized capacity of the machines, and the available machine capacity. Constraint (3.4) ensures that the number of machines does not exceed the limit in each cell. This constraint is practically important, since the high number of machines in the cell makes the control of the cell very difficult because of the increased intra cell material flow, which will decrease the effectiveness of the system. Constraint (3.5) ensures that each cell has an adequate number of operators to operate machines in the cell. For each cell, the equation consists of the demand for operator, unutilized capacity of operators, and the available operator time. Constraint (3.6) ensures that the number of operators

does not exceed the limit in the cell. Constraint (3.7) ensures that the frequency of lifting in each cell is in the appropriate range (for this case, smaller or equal to 3). Constraint (3.8) ensures that the composite lifting index does not exceed critical value (for this case, 1.5). Constraint (3.9) ensures the zero-one integer variables. Constraint (3.10) ensures the general integer variables. Constraint (3.11) ensures the non-negative decision variables.

3.1.4. Model 2: Single Period

3.1.4.1. Nomenclature

Indices:

p	=	parts index	$(p = 1, 2, \ldots\ldots, n_p)$
j	=	operation number	$(j = 1, 2, \ldots\ldots, n_j)$
m	=	machine types	$(m = 1, 2, \ldots\ldots, n_m)$
c	=	cells index	$(c = 1, 2, \ldots\ldots, n_c)$

Parameters:

D_p	=	average demand for part p per period, in units
F_m	=	amortized cost per period to procure one machine of type m
J_m	=	idle time cost per unit time of machine type m
t_{pjm}	=	unit processing time to perform operation j of part p on machine m
H_p	=	average handling cost of part p per intercellular move
G_m	=	proportion of operator attention required while one machine type m is operating
LW_p	=	load weight of part p, in kgs

LD_p = vertical lifting distance of part p, in cm = $|\, VK_p$ final – VK_p initial $|$

U = total shop time per period, in hours

S_c = maximum number of machines allowed in cell c

VW_c = maximum number of operators allowed in cell c

N = cost of one operator

JW = idle time cost of operator per unit time

LC = expected operator injury cost per operator per unit Composite Lifting Index

VK_{pc} = vertical distance of part p from knuckle in cell c (choose greater VK between initial and final condition), assumed to be 50 cm

$$L_{pj} = \begin{cases} 1 \text{ if } \sum_{m=1}^{n_m} t_{pjm} > 0 \text{ (there is operation performed)} \\ 0 \text{ if } \sum_{m=1}^{n_m} t_{pjm} = 0 \text{ (there is no operation performed)} \end{cases}$$

Decision variables:

0/1 integer variables:

$$X_{pjc} = \begin{cases} 1 & \text{if operation } j \text{ of part } p \text{ is assigned to cell } c \\ 0 & \text{otherwise} \end{cases}$$

$$V^+_{pjc} = \begin{cases} 1 & \text{if operation } j \text{ of part } p \text{ is performed in different cell than} \\ & \text{the preceding operation} \\ 0 & \text{otherwise} \end{cases}$$

General integer variables:

M_{mc} = number of machine type m assigned to cell c

W_c = number of operators assigned to cell c

General variables:

I_{mc} = idle time of machine type m assigned to cell c

WI_c = idle time of operator in cell c

FM_{pc} = frequency multiplier for the lifting frequency of part p in cell c

$STLI_{pjc}$ = single-task lifting index of operation j of part p in cell c

CLI_c = Composite Lifting Index of cell c

3.1.4.2. Mathematical Model

Minimize cost:

$$Z = \sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(F_m M_{mc} + J_m I_{mc}\right) + \sum_{c=1}^{n_c}\left(N \cdot U \cdot W_c + JW \cdot WI_c\right) + \sum_{p=1}^{n_p} H_p D_p \sum_{j=2}^{n_j}\sum_{c=1}^{n_c} V_{pjc}^{+} + LC\sum_{c=1}^{n_c} W_c CLI_c \tag{3.16}$$

Subject to:

$$\sum_{c=1}^{n_c} X_{pjc} = 1 \qquad \forall\, p \text{ and } j \tag{3.17}$$

$$I_{mc} + \sum_{p=1}^{n_p} D_p \sum_{j=1}^{n_j} t_{pjm} X_{pjc} = M_{mc} U \qquad \forall\, m \text{ and } c \tag{3.18}$$

$$\sum_{m=1}^{n_m} M_{mc} \le S_c \qquad \forall\, c \tag{3.19}$$

$$X_{pjc} - X_{p,j-1,c} = V_{pjc}^{+} - V_{pjc}^{-} \qquad \forall\, \mathrm{p}, j > 1, \text{ and } c \tag{3.20}$$

$$WI_c + \sum_{m=1}^{n_m} G_m \sum_{p=1}^{n_p} D_p \sum_{j=1}^{n_j} t_{pjm} X_{pjc} = W_c U \qquad \forall c \qquad (3.21)$$

$$W_c \le VW_c \qquad \forall c \qquad (3.22)$$

$$\sum_{p=1}^{n_p} \sum_{j=1}^{n_j} \left(\frac{D_p \cdot X_{pjc} \cdot L_{pj}}{U \cdot W_c} \right) \le 3 \qquad \forall c \qquad (3.23)$$

$$CLI_c \le 1.5 \qquad \forall c \qquad (3.24)$$

$$X_{pjc}, V^+_{pjc}, V^-_{pjc} \in \{0,1\} \qquad (3.25)$$

$$M_{mc}, W_c \in \text{ integer} \qquad (3.26)$$

$$M_{mc}, W_c, I_{mc}, WI_c \ge 0 \qquad (3.27)$$

Frequency Multiplier:

The Frequency Multiplier was adapted using linear regression with assumption that lifting is performed in a long duration. The detail of the linear regression is in Figure 2.1.

$$FM_{pjc} = 0.836 - 0.089 \left(\frac{D_p \cdot X_{pjc} \cdot L_{pj}}{U \cdot W_c} \right) \qquad (3.28)$$

Single-Task Lifting Index:

The Single-Task Lifting Index (*STLI*) is calculated based on the assumption that the horizontal multiplier, asymmetry multiplier, and coupling multiplier are equal to 1. The detail explanation of the *STLI* formula is in Chapter 2 (equation 2.11). Vertical multiplier, distance multiplier, and frequency multiplier are corresponding to equation 2.4, 2.5, and 2.6.

Equation 3.13: $$STLI = \frac{LW}{LC \times VM \times DM \times FM}$$

Equation 2.4: $VM = 1 - 0.003|V - KH| = 1 - 0.003VK$

Equation 2.5: $DM = 0.82 + {}^{4.5}/_{LD}$

Equation 2.6: $FM = 0.836 - 0.089F$

$$STLI_{pjc} = \frac{LW_p \cdot X_{pjc} \cdot L_{pj}}{23 \times \left[1 - 0.003VK_{pc}\right] \times \left[0.82 + {}^{4.5}/_{LD_p}\right] \times \left[0.836 - 0.089\left(\frac{D_p \cdot X_{pjc} \cdot L_{pj}}{U \cdot W_c}\right)\right]} \tag{3.29}$$

Composite Lifting Index (CLI):

The Composite Lifting Index formula below is an approximation from the original NIOSH formula (equation 2.12).

$$CLI_c = \max_{pj} STLI_{pjc} + 0.25 \times \left[\frac{\left(\sum_{p=1}^{n_p}\sum_{j=1}^{n_j} STLI_{pjc}\right) - \max_{pj} STLI_{pjc}}{\left(\sum_{p=1}^{n_p}\sum_{j=1}^{n_j} X_{pjc} L_{pj}\right) - 1}\right] \tag{3.30}$$

Objective function

The objective of the model is to minimize the various cost functions related to the design of cellular manufacturing systems using the NIOSH lifting index as the ergonomics measure for manual lifting tasks. There are six components considered in the objective function, which are: total machine procurement cost, total machine idle time cost, total operator salary cost, total operator idle time cost, total intercellular movements cost, and total lifting risk cost (total expected injury cost of manual lifting).

The first component in the objective function is the total machine procurement cost related with amortized cost per period to procure different types of machines in the system. The total machine procurement cost can be calculated by the summation of machine procurement cost multiplied by the number of machine for every machine type in every cell.

The second component is the total machine idle time cost which assigns a penalty cost on the unutilized capacity of the machine. The penalty for the idle time of the machine is defined as the opportunity cost of not processing the part on the machine which is equal to the loss of interest on the investment (for example, machine investment, space investment, and maintenance cost). Additionally, assigning penalty cost on the unutilized capacity of the machine will balance the utilization of the machines in the system without creating excess idle capacity. The total machine idle time cost can be calculated by the summation of machine idle time cost per unit time multiplied by the machine idle hours for every machine type in every cell.

The third component is the total operator cost related with operator assignment in the system. The total operator cost can be calculated by multiplying the cost of one operator per unit time by the operator's time per period, for all operators assigned to the system.

The fourth component is the total operator idle time cost which also assigns a penalty cost on the unutilized capacity of the operator. The penalty for the idle time of the operator is defined as the opportunity cost of not operating the machine or manual material handling which is equal to the loss of interest on the investment. The total

operator idle time cost can be calculated by the summation of operator idle time cost per unit time multiplied by the operator idle time in every cell.

The fifth component is the total intercellular movements cost which assigns a fixed amount of cost to each movement among cells. This cost may include several components, such as the operation and investment cost of the special material handling equipment, cost of special fixtures, and cost of controlling the increased material flow between cells. Even though allowing intercellular movements contradicts with the group technology philosophy, in practical situations it might be applicable as long as economically feasible. The total intercellular movements cost can be calculated by the summation of the intercellular movement cost multiplied by the number of intercellular movements that occur.

The sixth component is the total lifting risk cost (total expected injury cost) associated with manual lifting tasks. It is assumed that this cost is correlated with NIOSH lifting index. This cost can be calculated based on the expected injury cost rate per unit lifting index per operator times the lifting task index (NIOSH score) and the number of operators. This cost, as mentioned before, may include injury compensations, training costs for new operators to replace the injured operators, and productivity loss.

Constraints

The model developed in this section considers ten different operating constraint sets related to the cell formation design that has manual lifting tasks.

It was assumed that multiple units of the same machine can be assigned to one cell and each job on each part can only be allocated only to one cell. Constraint (3.17)

ensures this integrality by allocating each part type p in only one cell c. Constraint (3.18) ensures that each cell has an adequate number of machines of each type to perform its assigned workload. Constraint (3.19) ensures that the number of machines does not exceed the limit in each cell. Constraint (3.20) ensures the cost assignment of the intercellular movements. Each time a part makes a movement outside of the cell, the indicator variable V^{+}_{pjc} will have the value of 1 and then an intercellular movement cost will be assigned to the part. Constraint (3.21) ensures that each cell has an adequate number of operators to operate machines in the cell. Constraint (3.22) ensures that the number of operators does not exceed the limit in the cell. Constraint (3.23) ensures that the frequency of lifting in each cell is in the appropriate range (for this case, smaller or equal to 3). Constraint (3.24) ensures that the composite lifting index does not exceed critical value (for this case, 1.5). Constraint (3.25) ensures the zero-one integer variables. Constraint (3.26) ensures the general integer variables. Constraint (3.27) ensures the non-negative decision variables.

3.2. Mathematical Models under Multi Period Planning Horizon

3.2.1. Problem Definition

The mathematical models developed in this section attempt to form part-family and machine-grouping with operator assignment that does not exceed a certain limit of frequency of lifting and composite lifting index, for each period in the planning horizon. These mathematical models also balance the machining capacity and operator capacity distribution throughout the multi period planning horizon. Therefore, a relocation cost

will be assigned for each machine capacity change, and a manpower level change cost will be assigned for each operator capacity change.

3.2.2. Assumptions

1. Parts to be produced for the period are known and have constant demand throughout the period.
2. For each part, type of machines required and processing times on these machines are available.
3. For Model 3, each part is assigned only to one cell (no intercellular movements). For Model 4, each part is assigned based on the operation sequence (intercellular movements allowed).
4. For each part, the weights of lift for every operation are the same (constant weight).
5. Machines are in 100% capacity and various costs are known.
6. There can be multiple units of machines from each type in the system and cost of each machine is assumed to be the same for a given machine type.
7. Number of machine cells to be formed is determined before.
8. All machine cells are manual labor intensive (need operators to operate machines and to work on manual material handling).
9. Operator cost is the same for every operator assigned and operators are in 100% availability.
10. Expected injury cost per operator is known and calculated proportionally to the composite lifting index of the cell where the operator is assigned.

11. Lifting control is needed both in initial condition and final condition.
12. Frequency of lifting for one operator cannot exceed 3 lifts/min.
13. The Frequency Multiplier is adapted from Table 2.1 to a linear equation shown in Figure 2.1.
14. Lifting horizontal distance is below 25 cm, so that the Horizontal Multiplier is 1.
15. The highest vertical lifting distance from knuckle is 50 cm.
16. Vertical lifting distances for each part are known.
17. Lifting position is in the sagittal plane / neutral body posture / symmetrical condition, so that the Asymmetry Multiplier is 1.
18. The object is lifted in optimal container design with optimal handles that present a good coupling, so that the Coupling Multiplier is 1.
19. The composite lifting index for multi-task lifting is adapted from the rank algorithm to a formula.
20. The composite lifting index critical value is 1.5, which means the composite lifting index for each cell cannot exceed 1.5.
21. Changes in part family composition, machine cell configuration, and number of operators at the end of each period are allowed.
22. Inventory carrying to the next period is not allowed.
23. There is no production loss due to the changes in the machine cell configuration and number of operators.

3.2.3. Model 3: Multi Period

3.2.3.1. Nomenclature

Indices:

p = parts index ($p = 1, 2, \ldots\ldots, n_p$)

m = machine types ($m = 1, 2, \ldots\ldots, n_m$)

c = cells index ($c = 1, 2, \ldots\ldots, n_c$)

t = period index ($t = 1,2, \ldots\ldots, n_t$)

Parameters:

D_{pt} = average demand for part p in period t, in units

F_m = amortized cost per period to procure one machine of type m

J_m = idle time cost per unit time of machine type m

t_{pm} = unit processing time of part p on machine m

G_m = proportion of operator attention required while one machine type m is operating

LW_p = load weight of part p, in kgs

LD_p = vertical lifting distance of part p, in cm = $|\, VK_p$ final – VK_p initial $|$

U = total shop time per period, in hours

S_{ct} = maximum number of machines allowed in cell c in period t

VW_{ct} = maximum number of operators allowed in cell c in period t

N = cost of one operator

JW = idle time cost of operator per unit time

LC = expected operator injury cost per operator per unit Composite Lifting Index

VK_{pc} = vertical distance of part p from knuckle in cell c (choose greater VK between initial and final condition), assumed to be 50 cm

IM_{mt} = relocation cost of increasing number of machine m in the beginning of period t

DM_{mt} = relocation cost of decreasing number of machine m in the beginning of period t

IW_t = cost of increasing number of operators in the beginning of period t

DW_t = cost of decreasing number of operators in the beginning of period t

L_{pm} = $\begin{cases} 1 & \text{if } t_{pm} > 0 \text{ (there is operation performed in this machine)} \\ 0 & \text{if } t_{pm} = 0 \text{ (there is no operation performed)} \end{cases}$

Decision variables:

0/1 integer variables:

X_{pct} = $\begin{cases} 1 & \text{if part } p \text{ is assigned to cell } c \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$

General integer variables:

M_{mct} = number of machine type m assigned to cell c in period t

W_{ct} = number of operators assigned to cell c in period t

ΔM^{+}_{mct} = positive change (increasing) in the number of machine type m in cell c from period t-1 to t (at the beginning of period t)

ΔM^{-}_{mct} = negative change (decreasing) in the number of machine type m in cell c from period t-1 to t (at the beginning of period t)

ΔW^{+}_{ct} = positive change (increasing) in the number of operators in cell c from period t-1 to t

ΔW^{-}_{ct} = negative change (decreasing) in the number of operators in cell c from period t-1 to t

General variables:

I_{mct} = idle time of machine type m assigned to cell c in period t

WI_{ct} = idle time of operator in cell c in period t

FM_{pct} = frequency multiplier for the lifting frequency of part p in cell c in period t

$STLI_{pct}$ = single-task lifting index of part p in cell c in period t

CLI_{ct} = Composite Lifting Index of cell c in period t

3.2.3.2. Mathematical Model

Minimize cost:

$$Z = \sum_{t=1}^{n_t}\left(\sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(F_m M_{mct} + J_m I_{mct}\right) + \sum_{c=1}^{n_c}\left(N \cdot U \cdot W_{ct} + JW \cdot WI_{ct}\right) + LC\sum_{c=1}^{n_c} W_{ct} CLI_{ct}\right) + \sum_{t=2}^{n_t}\left(\sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(IM_{mt}\Delta M^{+}_{mct} + DM_{mt}\Delta M^{-}_{mct}\right) + \sum_{c=1}^{n_c}\left(IW_t \Delta W^{+}_{ct} + DW_t \Delta W^{-}_{ct}\right)\right) \quad (3.31)$$

Subject to:

$$\sum_{c=1}^{n_c} X_{pct} = 1 \qquad \forall\, p \text{ and } t \qquad (3.32)$$

$$I_{mct} + \sum_{p=1}^{n_p} D_{pt} t_{pm} X_{pct} = M_{mct} U \qquad \forall\, m, c, \text{ and } t \qquad (3.33)$$

$$\sum_{m=1}^{n_m} M_{mct} \le S_{ct} \qquad \forall\, c \text{ and } t \qquad (3.34)$$

$$WI_{ct} + \sum_{m=1}^{n_m} G_m \sum_{p=1}^{n_p} D_{pt} t_{pm} X_{pct} = W_{ct} U \qquad \forall\, c \text{ and } t \qquad (3.35)$$

$$W_{ct} \le VW_{ct} \qquad \forall\, c \text{ and } t \qquad (3.36)$$

$$\sum_{p=1}^{n_p} \left(\frac{X_{pct} \cdot D_{pt} \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_{ct}} \right) \le 3 \qquad \forall\, c \text{ and } t \qquad (3.37)$$

$$CLI_{ct} \le 1.5 \qquad \forall\, c \text{ and } t \qquad (3.38)$$

$$M_{mct} - M_{m,c,t-1} = \Delta M^{+}_{mct} - \Delta M^{-}_{mct} \qquad \forall\, m, c, \text{ and } t > 1 \qquad (3.39)$$

$$W_{ct} - W_{c,t-1} = \Delta W^{+}_{ct} - \Delta W^{-}_{ct} \qquad \forall\, c \text{ and } t > 1 \qquad (3.40)$$

$$X_{pct} \in \{0,1\} \qquad (3.41)$$

$$M_{mct}, W_{ct}, \Delta M^{+}_{mct}, \Delta M^{-}_{mct}, \Delta W^{+}_{ct}, \Delta W^{-}_{ct} \in \text{ integer} \qquad (3.42)$$

$$M_{mct}, W_{ct}, I_{mct}, WI_{ct}, \Delta M^{+}_{mct}, \Delta M^{-}_{mct}, \Delta W^{+}_{ct}, \Delta W^{-}_{ct} \ge 0 \qquad (3.43)$$

Frequency Multiplier:

The Frequency Multiplier was adapted using linear regression with assumption that lifting is performed in a long duration. The detail of the linear regression is in Figure 2.1.

$$FM_{pct} = 0.836 - 0.089\left(\frac{X_{pct} \cdot D_{pt} \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_{ct}}\right) \tag{3.44}$$

Single-Task Lifting Index:

The Single-Task Lifting Index (*STLI*) is calculated based on the assumption that the horizontal multiplier, asymmetry multiplier, and coupling multiplier are equal to 1. The detail explanation of the *STLI* formula is in Chapter 2 (equation 2.11). Vertical multiplier, distance multiplier, and frequency multiplier are corresponding to equation 2.4, 2.5, and 2.6.

Equation 3.13: $STLI = \dfrac{LW}{LC \times VM \times DM \times FM}$

Equation 2.4: $VM = 1 - 0.003|V - KH| = 1 - 0.003VK$

Equation 2.5: $DM = 0.82 + {}^{4.5}\!/_{LD}$

Equation 2.6: $FM = 0.836 - 0.089F$

$$STLI_{pct} = \frac{LW_p \cdot X_{pct} \cdot L_{pm}}{23 \times \left[1 - 0.003VK_{pc}\right] \times \left[0.82 + {}^{4.5}\!/_{LD_p}\right] \times \left[0.836 - 0.089\left(\dfrac{X_{pct} \cdot D_{pt} \sum_{m=1}^{n_m} L_{pm}}{U \cdot W_{ct}}\right)\right]} \tag{3.45}$$

Composite Lifting Index (CLI):

The Composite Lifting Index formula below is an approximation from the original NIOSH formula (equation 2.12).

$$CLI_{ct} = \max_{p} STLI_{pct} + 0.25 \times \left[\frac{\left(\sum_{p=1}^{n_p} STLI_{pct} \right) - \max_{p} STLI_{pct}}{\left(\sum_{p=1}^{n_p} X_{pct} \right) - 1} \right] \tag{3.46}$$

Objective function

The objective of the model is to minimize the various cost functions related to the design of cellular manufacturing systems using the NIOSH lifting index as the ergonomics measure for manual lifting tasks under multi period planning horizon. There are seven components considered in the objective function, which are: total machine procurement cost, total machine idle time cost, total operator salary cost, total operator idle time cost, total lifting risk cost (total expected injury cost of manual lifting), total machine relocation cost, and total manpower level change cost.

The first five components are modified from the Model 1 into multi period planning cost components. The additional cost components are the total machine relocation cost and total manpower level change cost.

The sixth component is the total machine relocation cost which is the cost charge for the changes in the configuration of the machines in the cell at the end of each period, as a result of the removal of existing machines or installation of new machines in each cell. The relocation cost (increasing or decreasing) may include installation or removal cost, cost of re-installing the cell system (for example, oil, water, or electrical system), and material handling systems. The total machine relocation cost can be calculated by the

summation of the machine change (increasing and decreasing) cost with the changes of every machine in every cell at the end of every period.

The seventh component is the total manpower level change cost which is the cost charge to change the allocation of operators in the cell at the end of each period. The total cost of increasing or decreasing of operators may include hiring cost, training cost, lay-off compensation, and so on.

Constraints

The model developed in this section considers ten different operating constraint sets related to the cell formation design that has manual lifting tasks.

It was assumed that multiple units of the same machine can be assigned to one cell and each part can be allocated only to one cell. Constraint (3.32) ensures this integrality by allocating each part type p in only one cell c. Constraint (3.33) ensures that each cell has an adequate number of machines of each type to perform its assigned workload. Constraint (3.34) ensures that the number of machines does not exceed the limit in each cell. Constraint (3.35) ensures that each cell has an adequate number of operators to operate machines in the cell. Constraint (3.36) ensures that the number of operators does not exceed the limit in the cell. Constraint (3.37) ensures that the frequency of lifting in each cell is in the appropriate range (for this case, smaller or equal to 3). Constraint (3.38) ensures that the composite lifting index does not exceed critical value (for this case, 1.5). Constraint (3.39) ensures the link among the planning periods, by keeping track of the changes in the number of machines in each cell from period to period. Constraint (3.40) ensures the link among the planning periods, by keeping track

of the changes in the number of operators in each cell from period to period. Constraint (3.41) ensures the zero-one integer variables. Constraint (3.42) ensures the general integer variables. Constraint (3.43) ensures the non-negative decision variables.

3.2.4. Model 4: Multi Period

3.2.4.1. Nomenclature

Indices:

p = parts index ($p = 1, 2, \ldots\ldots, n_p$)

j = operation number ($j = 1, 2, \ldots\ldots, n_j$)

m = machine types ($m = 1, 2, \ldots\ldots, n_m$)

c = cells index ($c = 1, 2, \ldots\ldots, n_c$)

t = period index ($t = 1,2, \ldots\ldots, n_t$)

Parameters:

D_{pt} = average demand for part p in period t, in units

F_m = amortized cost per period to procure one machine of type m

J_m = idle time cost per unit time of machine type m

t_{pjm} = unit processing time to perform operation j of part p on machine m

G_m = proportion of operator attention required while one machine type m is operating

LW_p = load weight of part p, in kgs

LD_p = vertical lifting distance of part p, in cm = $|\, VK_p$ final – VK_p initial $|$

U = total shop time per period, in hours

S_{ct} = maximum number of machines allowed in cell c in period t

VW_{ct} = maximum number of operators allowed in cell c in period t

N = cost of one operator

JW = idle time cost of operator per unit time

LC = expected operator injury cost per operator per unit Composite Lifting Index

VK_{pc} = vertical distance of part p from knuckle in cell c (choose greater VK between initial and final condition), assumed to be 50 cm

IM_{mt} = relocation cost of increasing number of machine m in the beginning of period t

DM_{mt} = relocation cost of decreasing number of machine m in the beginning of period t

IW_t = cost of increasing number of operators in the beginning of period t

DW_t = cost of decreasing number of operators in the beginning of period t

$$L_{pj} = \begin{cases} 1 \text{ if } \sum_{m=1}^{n_m} t_{pjm} > 0 \text{ (there is operation performed)} \\ 0 \text{ if } \sum_{m=1}^{n_m} t_{pjm} = 0 \text{ (there is no operation performed)} \end{cases}$$

Decision variables:

0/1 integer variables:

$$X_{pjct} = \begin{cases} 1 & \text{if operation } j \text{ of part } p \text{ is assigned to cell } c \text{ in period } t \\ 0 & \text{Otherwise} \end{cases}$$

$$V^{+}_{pjct} = \begin{cases} 1 & \text{if operation } j \text{ of part } p \text{ in period } t \text{ is performed in different cell than the preceding operation} \\ 0 & \text{otherwise} \end{cases}$$

General integer variables:

M_{mct} = number of machine type m assigned to cell c in period t

W_{ct} = number of operators assigned to cell c in period t

ΔM^{+}_{mct} = positive change (increasing) in the number of machine type m in cell c from period t-1 to t (at the beginning of period t)

ΔM^{-}_{mct} = negative change (decreasing) in the number of machine type m in cell c from period t-1 to t (at the beginning of period t)

ΔW^{+}_{ct} = positive change (increasing) in the number of operators in cell c from period t-1 to t

ΔW^{-}_{ct} = negative change (decreasing) in the number of operators in cell c from period t-1 to t

General variables:

I_{mct} = idle time of machine type m assigned to cell c in period t

WI_{ct} = idle time of operator in cell c in period t

FM_{pct} = frequency multiplier for the lifting frequency of part p in cell c in period t

$STLI_{pjct}$ = single-task lifting index of operation j of part p in cell c in period t

CLI_{ct} = Composite Lifting Index of cell c in period t

3.2.4.2. Mathematical Model

Minimize cost:

$$Z = \sum_{t=1}^{n_t}\left(\sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(F_m M_{mct} + J_m I_{mct}\right) + \sum_{c=1}^{n_c}\left(N \cdot U \cdot W_{ct} + JW \cdot WI_{ct}\right) + \sum_{p=1}^{n_p}\sum_{j=2}^{n_j}\sum_{c=1}^{n_c} H_p D_{pt} V^+_{pjct}\right.$$
$$\left. + LC\sum_{c=1}^{n_c} W_{ct} CLI_{ct}\right) + \sum_{t=2}^{n_t}\left(\sum_{m=1}^{n_m}\sum_{c=1}^{n_c}\left(IM_{mt}\Delta M^+_{mct} + DM_{mt}\Delta M^-_{mct}\right) + \sum_{c=1}^{n_c}\left(IW_t \Delta W^+_{ct} + DW_t \Delta W^-_{ct}\right)\right) \quad (3.47)$$

Subject to:

$$\sum_{c=1}^{n_c} X_{pjct} = 1 \qquad \forall\, p, j, \text{ and } t \quad (3.48)$$

$$I_{mct} + \sum_{p=1}^{n_p} D_{pt} \sum_{j=1}^{n_j} t_{pjm} X_{pjct} = M_{mct} U \qquad \forall\, m, c, \text{ and } t \quad (3.49)$$

$$\sum_{m=1}^{n_m} M_{mct} \le S_{ct} \qquad \forall\, c \text{ and } t \quad (3.50)$$

$$X_{pjct} - X_{p,j-1,c,t} = V^+_{pjct} - V^-_{pjct} \qquad \forall\, p, j > 1, c, \text{ and } t \quad (3.51)$$

$$WI_{ct} + \sum_{m=1}^{n_m} G_m \sum_{p=1}^{n_p} D_{pt} \sum_{j=1}^{n_j} t_{pjm} X_{pjct} = W_{ct} U \qquad \forall\, c \text{ and } t \quad (3.52)$$

$$W_{ct} \le VW_{ct} \qquad \forall\, c \text{ and } t \quad (3.53)$$

$$\sum_{p=1}^{n_p}\sum_{j=1}^{n_j}\left(\frac{D_{pt} \cdot X_{pjct} \cdot L_{pj}}{U \cdot W_{ct}}\right) \le 3 \qquad \forall\, c \text{ and } t \quad (3.54)$$

$$CLI_{ct} \le 1.5 \qquad \forall\, c \text{ and } t \quad (3.55)$$

$$M_{mct} - M_{m,c,t-1} = \Delta M^+_{mct} - \Delta M^-_{mct} \qquad \forall\, m, c, \text{ and } t > 1 \quad (3.56)$$

$$W_{ct} - W_{c,t-1} = \Delta W^+_{ct} - \Delta W^-_{ct} \qquad \forall\, c \text{ and } t > 1 \quad (3.57)$$

$$X_{pjct}, V^+_{pjct}, V^-_{pjct} \in \{0,1\} \quad (3.58)$$

$$M_{mct}, W_{ct}, \Delta M^+_{mct}, \Delta M^-_{mct}, \Delta W^+_{ct}, \Delta W^-_{ct} \in \text{ integer} \quad (3.59)$$

$$M_{mct}, W_{ct}, I_{mct}, WI_{ct}, \Delta M^{+}_{mct}, \Delta M^{-}_{mct}, \Delta W^{+}_{ct}, \Delta W^{-}_{ct} \geq 0 \quad (3.60)$$

Frequency Multiplier:

The Frequency Multiplier was adapted using linear regression with assumption that lifting is performed in a long duration. The detail of the linear regression is in Figure 2.1.

$$FM_{pjct} = 0.836 - 0.089\left(\frac{D_{pt} \cdot X_{pjct} \cdot L_{pj}}{U \cdot W_{ct}}\right) \quad (3.61)$$

Single-Task Lifting Index:

The Single-Task Lifting Index (*STLI*) is calculated based on the assumption that the horizontal multiplier, asymmetry multiplier, and coupling multiplier are equal to 1. The detail explanation of the *STLI* formula is in Chapter 2 (equation 2.11). Vertical multiplier, distance multiplier, and frequency multiplier are corresponding to equation 2.4, 2.5, and 2.6.

Equation 3.13: $$STLI = \frac{LW}{LC \times VM \times DM \times FM}$$

Equation 2.4: $$VM = 1 - 0.003|V - KH| = 1 - 0.003VK$$

Equation 2.5: $$DM = 0.82 + {}^{4.5}/_{LD}$$

Equation 2.6: $$FM = 0.836 - 0.089F$$

$$STLI_{pjct} = \frac{LW_p \cdot X_{pjct} \cdot L_{pj}}{23 \times \left[1 - 0.003VK_{pc}\right] \times \left[0.82 + {}^{4.5}/_{LD_p}\right] \times \left[0.836 - 0.089\left(\frac{D_{pt} \cdot X_{pjct} \cdot L_{pj}}{U \cdot W_{ct}}\right)\right]} \quad (3.62)$$

Composite Lifting Index (CLI):

The Composite Lifting Index formula below is an approximation from the original NIOSH formula (equation 2.12).

$$CLI_{ct} = \max_{pj} STLI_{pjct} + 0.25 \times \left[\frac{\left(\sum_{p=1}^{n_p} \sum_{j=1}^{n_j} STLI_{pjct} \right) - \max_{pj} STLI_{pjct}}{\left(\sum_{p=1}^{n_p} \sum_{j=1}^{n_j} X_{pjct} \cdot L_{pj} \right) - 1} \right] \quad (3.63)$$

Objective function

The objective of the model is to minimize the various cost functions related to the design of cellular manufacturing systems using the NIOSH lifting index as the ergonomics measure for manual lifting tasks under multi period planning horizon. There are eight components considered in the objective function, which are: total machine procurement cost, total machine idle time cost, total operator salary cost, total operator idle time cost, total intercellular movements cost, total lifting risk cost (total expected injury cost of manual lifting), total machine relocation cost, and total manpower level change cost.

The first six components are modified from the Model 2 into multi period planning cost components. The additional cost components are the total machine relocation cost and total manpower level change cost.

The seventh component is the total machine relocation cost which is the cost charge for the changes in the configuration of the machines in the cell at the end of each period, as a result of the removal of existing machines or installation of new machines in

each cell. The relocation cost (increasing or decreasing) may include installation or removal cost, cost of re-installing the cell system (for example, oil, water, or electrical system), and material handling systems. The total machine relocation cost can be calculated by the summation of the machine change (increasing and decreasing) cost with the changes of every machine in every cell at the end of every period.

The eighth component is the total manpower level change cost which is the cost charge to change the allocation of operators in the cell at the end of each period. The total cost of increasing or decreasing of operators may include hiring cost, training cost, lay-off compensation, and so on.

Constraints

The model developed in this section considers ten different operating constraint sets related to the cell formation design that has manual lifting tasks.

It was assumed that multiple units of the same machine can be assigned to one cell and each job on each part can be only allocated on a cell. Constraint (3.48) ensures this integrality by allocating each part type p in only one cell c. Constraint (3.49) ensures that each cell has an adequate number of machines of each type to perform its assigned workload. For each type of machine in the cell, the equation consists of the demand for capacity, unutilized capacity of the machine, and the available machine capacity. Constraint (3.50) ensures that the number of machines does not exceed the limit in each cell. Constraint (3.51) ensures the cost assignment of the intercellular movements. Constraint (3.52) ensures that each cell has an adequate number of operators to operate machines in the cell. Constraint (3.53) ensures that the number of operators does not

exceed the limit in the cell. Constraint (3.54) ensures that the frequency of lifting in each cell is in the appropriate range (for this case, smaller or equal to 3). Constraint (3.55) ensures that the composite lifting index does not exceed critical value (for this case, 1.5). Constraint (3.56) ensures the link among the planning periods, by keeping track of the changes in the number of machine in each cell from period to period. Constraint (3.57) also ensures the link among the planning periods, by keeping track of the changes in the number of operators in each cell from period to period. Constraint (3.58) ensures the zero-one integer variables. Constraint (3.59) ensures the general integer variables. Constraint (3.60) ensures the non-negative decision variables.

Chapter 4 –
Application of Genetic Algorithms

The mathematical models developed in Chapter 3 are mixed integer programming models. The decision variables consist of binary integer, general integer, and real number. The models also consist of nonlinear variables in the objective function and constraints, which means if solved using optimization software, it does not always give the global optimal solution or even cannot find a feasible solution. These nonlinear models work quite efficiently in terms of computational time for small size problems. However, as the problem size becomes larger, the computational time consumed to reach even a local optimal solution (time complexity) also increases exponentially due to the increase in the number of variables. It is experienced in some cases that the optimization software cannot find a feasible solution. Therefore, it is considered beneficial to develop a heuristic technique to deal with the complexity of nonlinear models and to reduce the amount of computational time needed to solve large size problems.

In this chapter, the application of genetic algorithms as a heuristic technique is developed to find optimal or near-optimal solutions for the mathematical models.

4.1. Introduction to Genetic Algorithms

Genetic Algorithms (GAs) are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string

structures (chromosomes) with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old (an occasional new part is tried for good measure). The chromosomes with poor fitness value (objective function value of the optimization problem) are discarded from the population at the end of each generation, which implies that fitness values of the chromosomes remaining in the population will either improve of remain the same from generation to generation. While randomized, GAs are no simple random walk. They efficiently exploit historical information to speculate on new search points with expected improved performance.

GAs have been developed by John Holland, his colleagues, and his student at the University of Michigan. The goals of their research have been twofold: (1) to abstract and rigorously explain the adaptive processes of natural systems, and (2) to design artificial system software that retains the important mechanisms of natural systems. This approach has led to important discoveries in both natural and artificial systems science.

In order for GAs to work more efficiently than traditional optimization method, GAs must differ in some very fundamental ways. GAs are different from more normal optimization and search procedures in four ways (Goldberg, 1989):

1. GAs work with a coding of the parameter set, not the parameters themselves.
2. GAs search from population of points, not a single point.
3. GAs use payoff (objective function) information, not derivatives or other auxiliary knowledge.
4. GAs use probabilistic transition rules, not deterministic rules.

4.2. General Structure of Genetic Algorithms

Genetic Algorithms, differing from conventional search techniques, start with an initial set of random solutions called *population*. Each individual in the population is called a *chromosome*, representing a solution to the problem at hand. A chromosome is a string of symbols; it is usually but not necessarily, a binary bit string. The chromosomes *evolve* through successive iterations, called *generations*. During each generation, the chromosomes are *evaluated*, using some measures of *fitness*. To create the next generation, new chromosomes, called *offspring*, are formed by either (a) merging two chromosomes from current generation using a *crossover* operator, or (b) modifying a chromosome using a *mutation* operator. A new generation is formed by (a) selecting, according to the fitness value, some of the parents and offspring and (b) rejecting others so as to keep the population size constant. Fitter chromosomes have higher probabilities of being selected. After several generations, the algorithms converge to the best chromosome, which hopefully represents the optimal or near-optimal solution to the problem.

In fact, there are only two kinds of operations in GAs (Gen and Cheng, 1997):

1. Genetic operations: ***Crossover*** and ***Mutation***
2. Evolution operation: ***Selection*** or ***Reproduction***

The genetic operations mimic the process of heredity of genes to create new offspring at each generation. The evolution operation mimics the process of *Darwinian evolution* to create populations from generation to generation.

4.2.1. Genetic Operator: Crossover

Crossover is the main genetic operator. It operates on two chromosomes at a time and generates offspring by combining both chromosomes' features. A simple way to achieve crossover would be to choose a random cut-point and generate offspring by combining the segment of one parent to the left of the cut-point with the segment of the other parent to the right of the cut-point. This method works well with the bit string representation. The performance of genetic algorithms depends, to a great extent, on the performance of the crossover operator used.

The *crossover rate* (denoted by p_c) is defined as the ratio of the number of offspring produced in each generation to the population size (usually denoted by *pop_size*). This ratio controls the expected number $p_c \times pop\_size$ of chromosomes to undergo the crossover operation. A higher crossover rate allows exploration of more of the solution space and reduces the chances of settling for a false optimum; but if this rate is too high, it results in the wastage of a lot of computation time in exploring unpromising regions of the solution space.

4.2.2. Genetic Operator: Mutation

Mutation is a background operator which produces spontaneous random changes in various chromosomes. A simple way to achieve mutation would be to alter one or more genes. In genetic algorithms, mutation serves the crucial role of either (a) replacing the genes lost from the population during the selection process so that they can be tried in a new context, or (b) providing the genes that were not present in the initial population.

The *mutation rate* (denoted by p_m) is defined as the percentage of the total number of genes in the population. The mutation rate controls the rate at which new genes are introduced into the population for trial. If it is too low, many genes that would have been useful are never tried out; but if it is too high, there will be much random perturbation, the offspring will start losing their resemblance to the parents, and the algorithm will lose the ability to learn from the history of search.

4.2.3. Evolution Operator: Selection or Reproduction

Selection or reproduction is a process in which chromosomes for the next generation are chosen from the members of the population at current generation by a probabilistic selection process. Population members with better fitness value (objective function in mathematical models) thus have a higher probability of contributing one or more offspring to the next generation. This operator can be seen as an artificial version of natural selection, a Darwinian survival of the fittest among the chromosomes.

4.3. Genetic Algorithm Development under Single Period Planning Horizon

In this section, genetic algorithms will be applied to the mathematical models developed under single period planning horizon in order to find an optimal or near-optimal solution. The algorithms will simultaneously develop the part families, machine cells, and the operator assignment. All the assumptions, objective function, cost components, and operating constraints stated for mathematical Model 1 and Model 2, are valid for the genetic algorithm applications. Section 4.3.1 explains the development of Genetic Algorithm for Model 1 (GA1), and Section 4.3.2 explains the development of

Genetic Algorithm for Model 2 (GA2). The computer code for the algorithm is in Appendix II.

4.3.1. Genetic Algorithm for Model 1 (GA1)

GA1 codes the optimization problem in Model 1 as a chromosome by using integer numbers where each *gene*, each position in the chromosome, correspond to the cell number which one part is assigned. Simultaneously, according to the chromosome part families, machine cells, and operator assignment are formed. Each chromosome has a length equal to total number of parts specified by the decision maker. Then the value of a gene is ranged from 1 to the total number of cells. For example, for 10 different parts and 3 machine cells arrangement, the length of chromosome is 10 and the value of genes is ranged from 1 to 3. This method of developing the chromosome gives an advantage to help the constraint 1.a (one part can only be assigned to one cell), since each value of gene in the chromosome can only be one integer which equal to the cell number. Figure 4.1 illustrates this example in the chromosome design used in GA1.

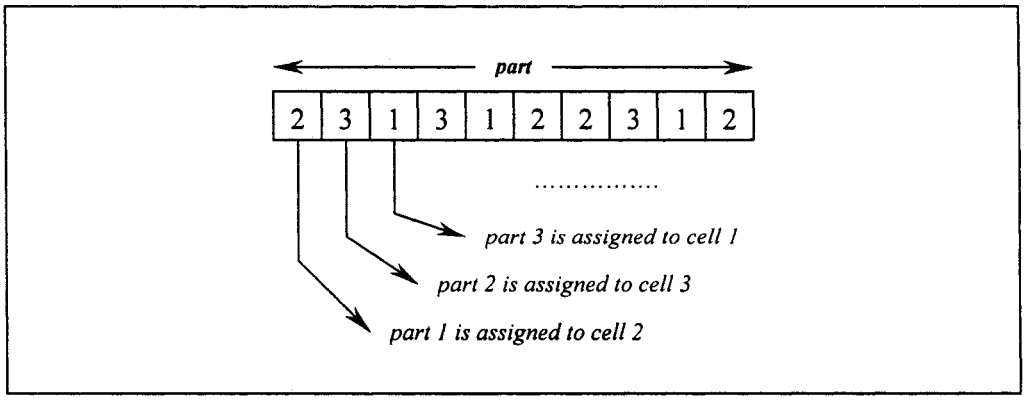

Figure 4.1: Illustration of the Chromosome representation for GA1

GA1 defines the population (set of chromosomes) as an array records. Each individual record, which contains a possible solution to the optimization problem, contains three attributes, which are:

1. Chromosome, which is the coding of the optimization problem as a string structure.
2. Fitness value, which is the resulting objective function value calculated by using the cost components of the model.
3. Probability, which is the records probability of giving offspring to the next generation, also defined as the probability of being selected to the next generation. The probability is calculated based on the fitness value of the chromosome.

The algorithm operates by calling several procedures, which can be summarized as follows:

1. Procedure GENERATE: Creates the initial population randomly. The string for the population is described before in Figure 4.1.
2. Procedure EVALUATION: Evaluates the fitness value of each string record in the population.
3. Procedure SELECTION: Creates a new population by selecting among the old population and copying them into the new population.
4. Procedure CROSSOVER: Creates two new string records, by randomly selecting two strings from the current population and mating their string structures.
5. Procedure MUTATION: Creates new string record, by altering the value of gene or genes in one randomly selected string structure.
6. Procedure SUMMARY: Summarizes the solution data.

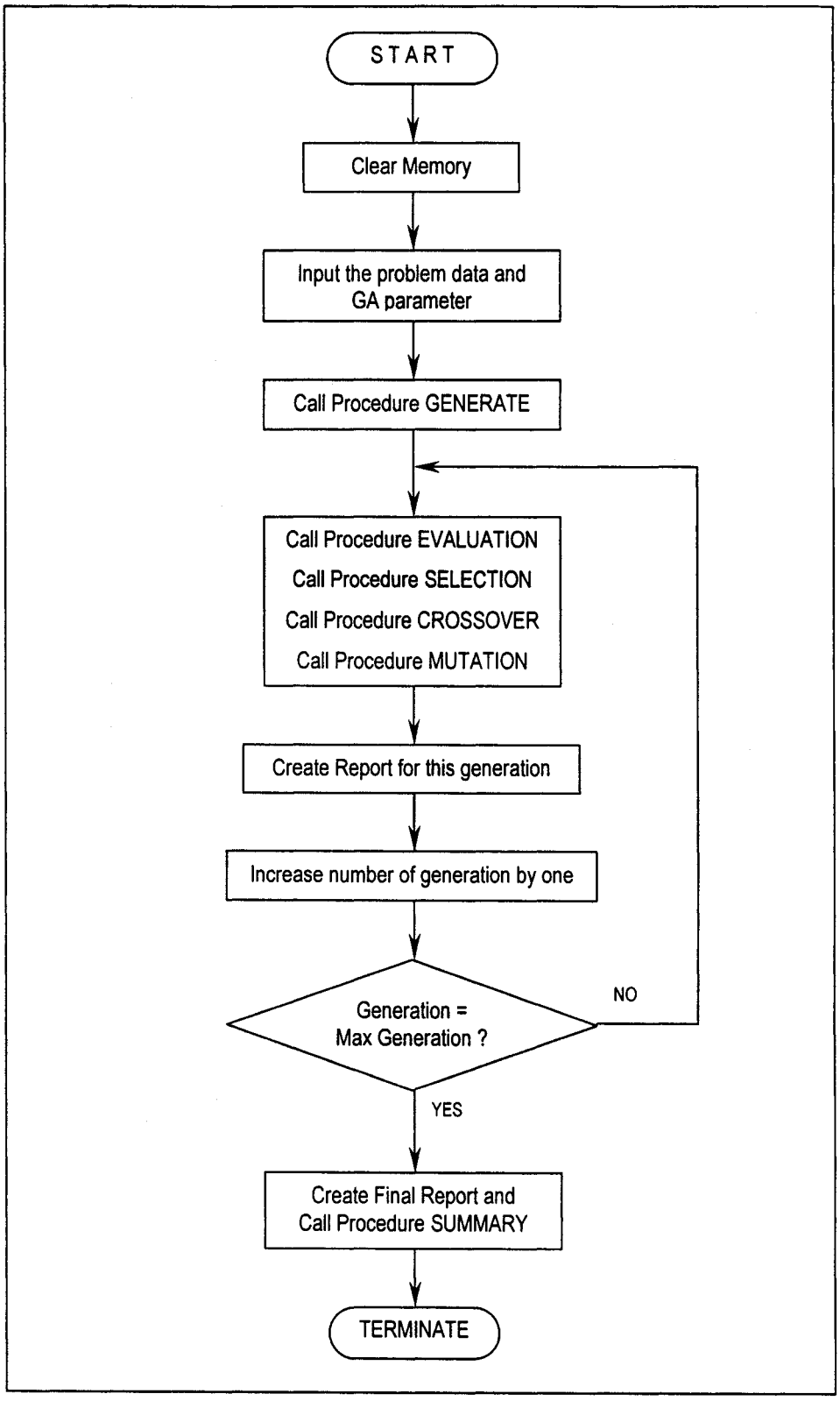

Figure 4.2: Flow Chart of Genetic Algorithms

Main Algorithm (GA1)

1. Clear memory.
2. Input the problem data: parts information, machines information, cells information, processing time of parts on machines, and other information (total time in a period, operator cost, operator idle time cost, lifting risk compensation).
3. Set the GA parameter:
    - population size: default is the number of parts,
    - maximum number of generation: default is number of cells multiplied with 1500,
    - probability of crossover: default is 0.80,
    - probability of mutation: default is 0.08.
4. Call the Procedure GENERATE, to create string structures for the initial population.
5. Call the Procedure EVALUATION, to evaluate the objective function and fitness value of each string structures, and also calculate the probability of being selected to next generation.
6. Call the Procedure SELECTION, to create the new population by selecting among the eligible records in the old population.
7. Call the Procedure CROSSOVER, to create two new string structures by crossover.
8. Call the Procedure MUTATION, to create new string structure by mutation.
9. Create report for this generation by recording the best objective function and the average objective function value of the population, and also updating the solution data of best objective function.
10. Increase the number of Generation by one.

11. If the Generation = Max Generation then go to step 11, otherwise go to step 5.
12. Create Final Report by finding the best fitness value among the generation record and recording the solution data.
13. Call the Procedure SUMMARY, to create the summary of the best solution.
14. Terminate.

Procedure GENERATE

1. Input the GA parameter, number of parts, number of machines, and number of cells from the Main Algorithm.
2. For each part, generate random integer number to assign part to cell, range between 1 and number of cells.
3. Increase the number of Population by one.
4. If the Population = Population Size then go to step 5, otherwise go to step 2.
5. Return the value of initial population to the Main Algorithm.
6. Terminate.

Procedure EVALUATION

1. Input the string structures, GA parameter, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration to start evaluation.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.

5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
7. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
8. Calculate the costs: machine cost, machine idle time cost, operator cost, operator idle time cost, and lifting risk cost.
9. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
10. Calculate the objective function by the summation of all cost components and the penalty.
11. Increase the number of Population by one.
12. If the Population = Population Size then go to step 13, otherwise go to step 3.
13. Calculate the fitness value of each string structure. The fitness values can be calculated by this formula:

$$FitnessValue_s = \frac{d/1000}{(d/1000) + ObjFunction_s - d} \tag{4.1}$$

where d = the minimum objective function from the population.

14. Calculate the probability of being selected to next generation and the cumulative probability.
15. Return the machine cell formation & operator assignment, objective functions, costs & penalties, and string probabilities & cumulative probabilities to the Main Algorithm.
16. Terminate.

Procedure SELECTION

1. Input the string structures, GA parameter, and string probabilities & cumulative probabilities.
2. Create random real number between 0 and 1.
3. Find in which string cumulative probability range that the random number fit, and reproduce the string structure to next generation (also known as the Roulette Wheel Selection).
4. Increase the number of Population by one.
5. If the Population = Population Size then go to step 6, otherwise go to step 2.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure CROSSOVER

1. Input the string structures and GA parameter.

2. For each string structures, create random real number between 0 and 1, and if the random number ≤ probability of crossover then put the string number in Crossover array, otherwise skip.
3. If the length of Crossover array is odd then go back to step 2, otherwise go to step 4.
4. Change the string structures in the Crossover array to pairs of string and put the pairs in the CPair array.
5. For each pair of crossover strings, create random integer number between 1 and 3 to decide the crossover type. Crossover types include 1-cut-point crossover, 2-cut-point crossover, and 3-cut-point crossover.
6. For each pair of crossover string, assign the cut-points based on the crossover type. The cut-points are random integer numbers between 1 and number of parts.
7. For each pair of crossover strings, do crossover to get new generation.
8. Return the new generation of string structures to the Main Algorithm.
9. Terminate.

Procedure MUTATION

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number ≤ probability of mutation then put the string number in Mutation array, otherwise skip.
3. For each mutation strings, create random integer number between 1 and 3 to decide the mutation type. Mutation types include self mutation, switch mutation, and copy mutation.

4. For each mutation string, assign the mutation-points based on the mutation type. The mutation-points are random integer numbers between 1 and number of parts.
5. For each mutation strings, do mutation to get new generation.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure SUMMARY

1. Input the string structure with best objective function, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
7. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
8. Calculate the costs: machine cost, machine idle time cost, operator cost, operator idle time cost, and lifting risk cost.
9. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of

machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.

10. Calculate the objective function by the summation of all cost components and the penalty.
11. Return the machine and operator assignment, total machine cost, total machine idle time cost, total operator cost, total operator idle time cost, total lifting risk cost, frequency of lifting in each cell, and composite lifting index in each cell to the Main Algorithm.
12. Terminate.

Final Report

The Final Report from the Genetic Algorithm (GA1) consists of:

1. Part Families (SolF): The string arrangement is mentioned before in Figure 4.1.
2. Machine Groups and Operator Assignments (SolG):

   The string arrangement for SolG is presented in Figure 4.3.
3. Frequency of Lifting (FoL): Array with size 1 × Total Machine Cells.
4. Composite Lifting Index (CLI): Array with size 1 × Total Machine Cells.
5. Cost components: Machine Cost (MC), Machine Idle Time Cost (MIC), Operator Cost (WC), Operator Idle Time Cost (WIC), and Lifting Risk Cost (LC).

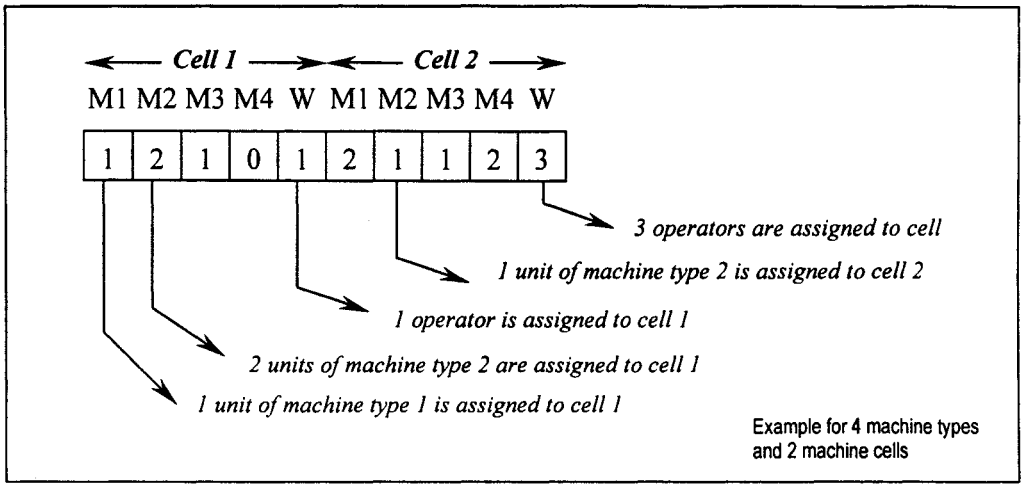

Figure 4.3: Machine Groups and Operator Assignments array

4.3.2. Genetic Algorithm for Model 2 (GA2)

GA2 codes the optimization problem in Model 2 as a chromosome by using integer numbers where each *gene*, each position in the chromosome, correspond to the cell number which one job or operation of one part is assigned. Simultaneously, according to the chromosome part families, machine cells, and operator assignment are formed. Each chromosome has a length equal to total number of parts multiplied with total number of maximum operation on each part. Then the value of a gene is ranged from 1 to the total number of cells. For example, for 10 different parts, 3 operations or less, and 3 machine cells arrangement, the length of chromosome is 30 and the value of genes is ranged from 1 to 3. This method of developing the chromosome gives an advantage to help the constraint set 2.a (one operation can only be assigned to one cell), since each value of gene in the chromosome can only be one integer which equal to the cell number. Figure 4.4 illustrates this example in the chromosome design used in GA2.

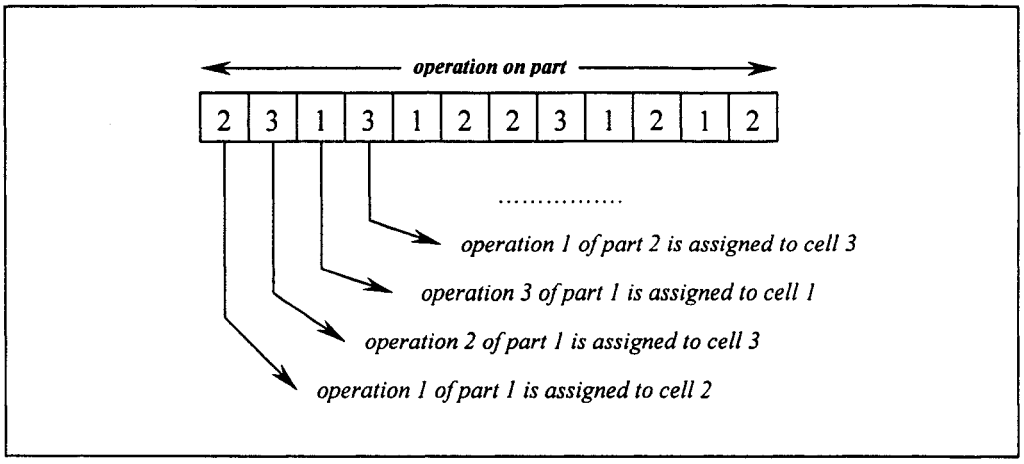

Figure 4.4: Chromosome representation for GA2

The algorithm operates by calling several procedures, which are the same with GA1.

Main Algorithm (GA2)

1. Clear memory.
2. Input the problem data: parts information, machines information, cells information, processing time of each part operation of each part on machines, and other information (total time in a period, operator cost, operator idle time cost, lifting risk compensation).
3. Set the GA parameter:
    - population size: default is the largest integer obtained from the number of parts multiplied with the maximum number of operations and divided with 2,
    - maximum number of generation: default is number of cells multiplied with 1500,
    - probability of crossover: default is 0.80,
    - probability of mutation: default is 0.08.

4. Call the Procedure GENERATE, to create string structures for the initial population.
5. Call the Procedure EVALUATION, to evaluate the objective function and fitness value of each string structures, and also calculate the probability of being selected to next generation.
6. Call the Procedure SELECTION, to create the new population by selecting among the eligible records in the old population.
7. Call the Procedure CROSSOVER, to create two new string structures by crossover.
8. Call the Procedure MUTATION, to create new string structure by mutation.
9. Create report for this generation by recording the best objective function and the average objective function value of the population, and also updating the solution data of best objective function.
10. Increase the number of Generation by one.
11. If the Generation = Max Generation then go to step 11, otherwise go to step 5.
12. Create Final Report by finding the best fitness value among the generation record and recording the solution data.
13. Call the Procedure SUMMARY, to create the summary of the best solution.
14. Terminate.

Procedure GENERATE

1. Input the GA parameter, number of parts, number of machines, and number of cells from the Main Algorithm.
2. For each operation on each part, generate random integer number to assign part to cell, range between 1 and number of cells.

3. Increase the number of Population by one.
4. If the Population = Population Size then go to step 5, otherwise go to step 2.
5. Return the value of initial population to the Main Algorithm.
6. Terminate.

Procedure EVALUATION

1. Input the string structures, GA parameter, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration to start evaluation.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the number of intercellular movements for each part.
7. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
8. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
9. Calculate the costs: machine cost, machine idle time cost, intercellular movements cost, operator cost, operator idle time cost, and lifting risk cost.
10. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of

machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.

11. Calculate the objective function by the summation of all cost components and the penalty.
12. Increase the number of Population by one.
13. If the Population = Population Size then go to step 13, otherwise go to step 3.
14. Calculate the fitness value of each string structure. The fitness values can be calculated using the Equation 4.1.
15. Calculate the probability of being selected to next generation and the cumulative probability.
16. Return the machine cell formation & operator assignment, objective functions, costs & penalties, and string probabilities & cumulative probabilities to the Main Algorithm.
17. Terminate.

Procedure SELECTION

1. Input the string structures, GA parameter, and string probabilities & cumulative probabilities.
2. Create random real number between 0 and 1.

3. Find in which string cumulative probability range that the random number fit, and reproduce the string structure to next generation (also known as the Roulette Wheel Selection).
4. Increase the number of Population by one.
5. If the Population = Population Size then go to step 6, otherwise go to step 2.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure CROSSOVER

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number ≤ probability of crossover then put the string number in Crossover array, otherwise skip.
3. If the length of Crossover array is odd then go back to step 2, otherwise go to step 4.
4. Change the string structures in the Crossover array to pairs of string and put the pairs in the CPair array.
5. For each pair of crossover strings, create random integer number between 1 and 3 to decide the crossover type. Crossover types include 1-cut-point crossover, 2-cut-point crossover, and 3-cut-point crossover.
6. For each pair of crossover string, assign the cut-points based on the crossover type. The cut-points are random integer numbers between 1 and number of parts multiplied with maximum number of operation.
7. For each pair of crossover strings, do crossover to get new generation.

8. Return the new generation of string structures to the Main Algorithm.
9. Terminate.

Procedure MUTATION

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number $\leq$ probability of mutation then put the string number in Mutation array, otherwise skip.
3. For each mutation strings, create random integer number between 1 and 3 to decide the mutation type. Mutation types include self mutation, switch mutation, and copy mutation.
4. For each mutation string, assign the mutation-points based on the mutation type. The mutation-points are random integer numbers between 1 and number of parts multiplied with maximum number of operation.
5. For each mutation strings, do mutation to get new generation.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure SUMMARY

1. Input the string structure with best objective function, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration.

3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the number of intercellular movements for each part.
7. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
8. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
9. Calculate the costs: machine cost, machine idle time cost, intercellular movements cost, operator cost, operator idle time cost, and lifting risk cost.
10. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
11. Calculate the objective function by the summation of all cost components and the penalty.
12. Return the machine and operator assignment, total machine cost, total machine idle time cost, total intercellular movements cost, total operator cost, total operator idle

time cost, total lifting risk cost, frequency of lifting in each cell, and composite lifting index in each cell to the Main Algorithm.

13. Terminate.

Final Report

The Final Report from the Genetic Algorithm (GA2) consists of:

1. Part Families (SolF): The string arrangement is mentioned before in Figure 4.4.
2. Machine Groups and Operator Assignments (SolG):

   The string arrangement for SolG is presented in Figure 4.5.
3. Frequency of Lifting (FoL): Array with size 1 × Total Machine Cells.
4. Composite Lifting Index (CLI): Array with size 1 × Total Machine Cells.
5. Cost components: Machine Cost (MC), Machine Idle Time Cost (MIC), Operator Cost (WC), Operator Idle Time Cost (WIC), Intercellular Movements Cost (IMC), and Lifting Risk Cost (LC).

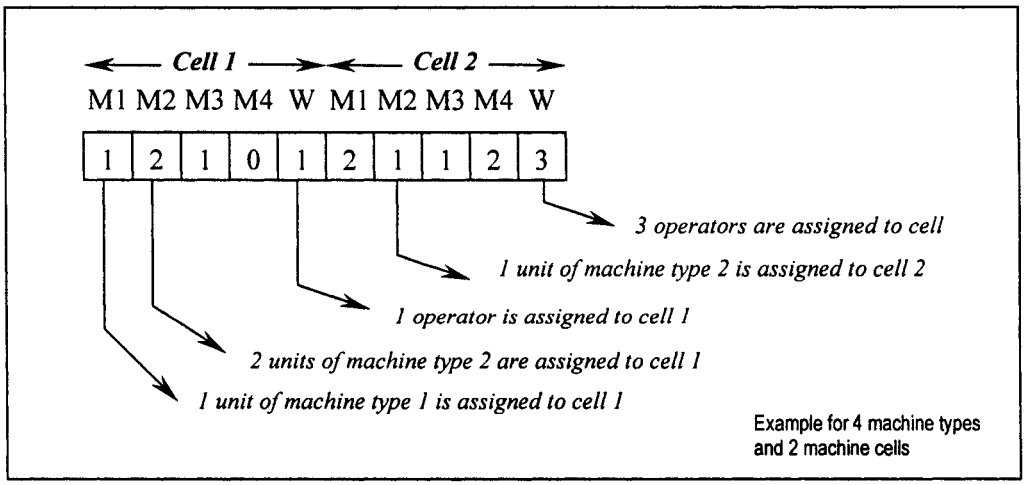

Figure 4.5: Machine Groups and Operator Assignments array

4.4. Genetic Algorithm Development under Multi Period Planning Horizon

In this section, genetic algorithms will be applied to the mathematical models developed under multi period planning horizon in order to find an optimal or near-optimal solution. The algorithms will simultaneously develop the part families, machine cells, and the operator assignment. All of the assumptions, objective function, cost components, and operating constraints stated for mathematical Model 3 and Model 4, are valid for the genetic algorithm applications. Section 4.4.1 explains the development of Genetic Algorithm for Model 3 (GA3), and Section 4.4.2 explains the development of Genetic Algorithm for Model 4 (GA4). The computer code for the algorithm is in Appendix II.

4.4.1. Genetic Algorithm for Model 3 (GA3)

GA3 codes the optimization problem in Model 3 as a chromosome by using integer numbers where each *gene*, each position in the chromosome, correspond to the cell number which one part is assigned. Simultaneously, according to the chromosome part families, machine cells, and operator assignment are formed. Each chromosome has a length equal to total number of parts, and has a number of rows equal to total number of planning period. Then the value of a gene is ranged from 1 to the total number of cells. For example, for 10 different parts and 3 machine cells arrangement for 2 planning periods, the length of chromosome is 10, the number of rows is 2, and the value of genes is ranged from 1 to 3. This method of developing the chromosome gives an advantage to help the constraint set 3.a (one operation can only be assigned to one cell), since each

value of gene in the chromosome can only be one integer which equal to the cell number. Figure 4.6 illustrates this example in the chromosome design used in GA3.

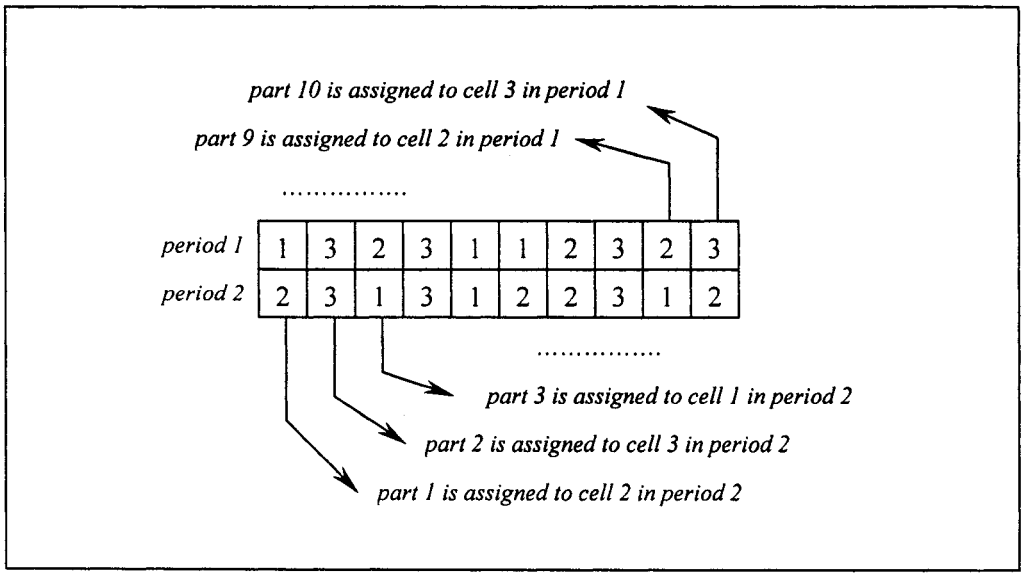

Figure 4.6: Chromosome representation for GA3

The algorithm operates by calling several procedures, which are also the same with GA1.

Main Algorithm (GA3)

1. Clear memory.
2. Input the problem data: parts information, machines information, cells information, processing time of parts on machines, and other information (total time in a period, operator cost, operator idle time cost, lifting risk compensation).
3. Set the GA parameter:
    - population size: default is the number of parts,
    - maximum number of generation: default is number of cells multiplied with 1500,

- probability of crossover: default is 0.80,
- probability of mutation: default is 0.08.

4. Call the Procedure GENERATE, to create string structures for the initial population.
5. Call the Procedure EVALUATION, to evaluate the objective function and fitness value of each string structures, and also calculate the probability of being selected to next generation.
6. Call the Procedure SELECTION, to create the new population by selecting among the eligible records in the old population.
7. Call the Procedure CROSSOVER, to create two new string structures by crossover.
8. Call the Procedure MUTATION, to create new string structure by mutation.
9. Create report for this generation by recording the best objective function and the average objective function value of the population, and also updating the solution data of best objective function.
10. Increase the number of Generation by one.
11. If the Generation = Max Generation then go to step 11, otherwise go to step 5.
12. Create Final Report by finding the best fitness value among the generation record and recording the solution data.
13. Call the Procedure SUMMARY, to create the summary of the best solution.
14. Terminate.

Procedure GENERATE

1. Input the GA parameter, number of parts, number of machines, and number of cells from the Main Algorithm.

2. For each part in each period, generate random integer number to assign part to cell, range between 1 and number of cells.
3. Increase the number of Population by one.
4. If the Population = Population Size then go to step 5, otherwise go to step 2.
5. Return the value of initial population to the Main Algorithm.
6. Terminate.

Procedure EVALUATION

1. Input the string structures, GA parameter, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration to start evaluation.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
7. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
8. Calculate the costs: machine cost, machine idle time cost, operator cost, operator idle time cost, and lifting risk cost.
9. Increase the number of Period by one.

10. If the Period = Total Number of Period then go to step 11, otherwise go to step 3.
11. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
12. Calculate the objective function by the summation of all cost components and the penalty.
13. Increase the number of Population by one.
14. If the Population = Population Size then go to step 15, otherwise go to step 3.
15. Calculate the fitness value of each string structure. The fitness values can be calculated using the Equation 4.1.
16. Calculate the probability of being selected to next generation and the cumulative probability.
17. Return the machine cell formation & operator assignment, objective functions, costs & penalties, and string probabilities & cumulative probabilities to the Main Algorithm.
18. Terminate.

Procedure SELECTION

1. Input the string structures, GA parameter, and string probabilities & cumulative probabilities.

2. Create random real number between 0 and 1.
3. Find in which string cumulative probability range that the random number fit, and reproduce the string structure to next generation (also known as the Roulette Wheel Selection).
4. Increase the number of Population by one.
5. If the Population = Population Size then go to step 5, otherwise go to step 2.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure CROSSOVER

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number $\leq$ probability of crossover then put the string number in Crossover array, otherwise skip.
3. If the length of Crossover array is odd then go back to step 2, otherwise go to step 4.
4. Change the string structures in the Crossover array to pairs of string and put the pairs in the CPair array.
5. For each pair of crossover strings, create random integer number between 1 and 3 to decide the crossover type. Crossover types include 1-cut-point crossover, 2-cut-point crossover, and 3-cut-point crossover.
6. For each pair of crossover string, assign the cut-points based on the crossover type. The cut-points are random integer numbers between 1 and number of parts.
7. For each pair of crossover strings, do crossover to get new generation.

8. Return the new generation of string structures to the Main Algorithm.
9. Terminate.

Procedure MUTATION

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number $\leq$ probability of mutation then put the string number in Mutation array, otherwise skip.
3. For each mutation strings, create random integer number between 1 and 3 to decide the mutation type. Mutation types include self mutation, switch mutation, and copy mutation.
4. For each mutation string, assign the mutation-points based on the mutation type. The mutation-points are random integer numbers between 1 and number of parts.
5. For each mutation strings, do mutation to get new generation.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure SUMMARY

1. Input the string structure with best objective function, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.

4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
7. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
8. Calculate the costs: machine cost, machine idle time cost, operator cost, operator idle time cost, and lifting risk cost.
9. Increase the number of Period by one.
10. If the Period = Total Number of Period then go to step 11, otherwise go to step 3.
11. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
12. Calculate the objective function by the summation of all cost components and the penalty.
13. Return the machine and operator assignment, total machine cost, total machine idle time cost, total operator cost, total operator idle time cost, total lifting risk cost, frequency of lifting in each cell, and composite lifting index in each cell to the Main Algorithm.

14. Terminate.

Final Report

The Final Report from the Genetic Algorithm (GA3) consists of:

1. Part Families (SolF): The string arrangement is mentioned before in Figure 4.6.
2. Machine Groups and Operator Assignments (SolG):

   The string arrangement for SolG is presented in Figure 4.7.
3. Frequency of Lifting (FoL): Array with size Total Period × Total Machine Cells.
4. Composite Lifting Index (CLI): Array with size Total Period × Total Machine Cells.
5. Cost components: Machine Cost (MC), Machine Idle Time Cost (MIC), Operator Cost (WC), Operator Idle Time Cost (WIC), Lifting Risk Cost (LC), Machine Relocation Cost (MCCost), and Manpower Level Change Cost (WCCost).

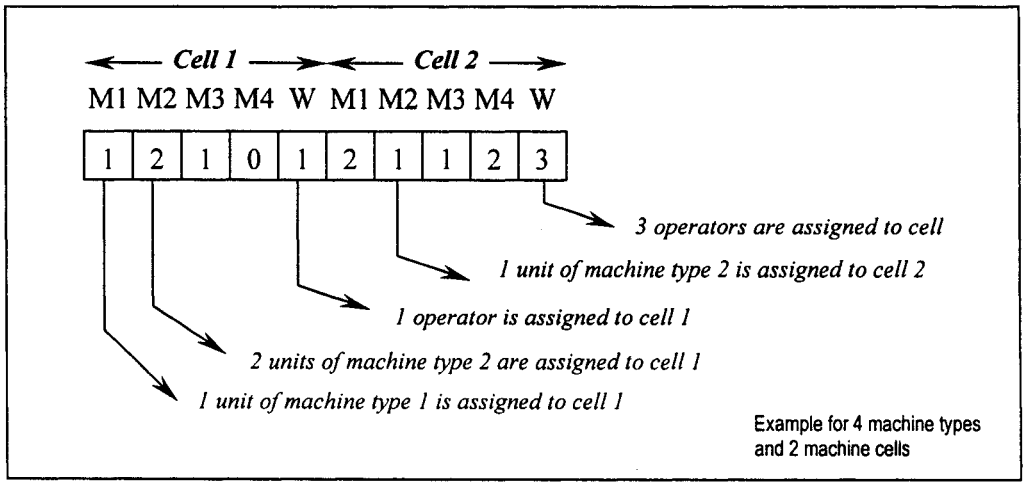

Figure 4.7: Machine Groups and Operator Assignments array

4.4.2. Genetic Algorithm for Model 4 (GA4)

GA4 codes the optimization problem in Model 4 as a chromosome by using integer numbers where each *gene*, each position in the chromosome, correspond to the

cell number which one part is assigned. Simultaneously, according to the chromosome part families, machine cells, and operator assignment are formed. Each chromosome has a length equal to total number of parts multiplied with the maximum number of operations, and has a number of rows equal to total number of planning period. Then the value of a gene is ranged from 1 to the total number of cells. For example, for 10 different parts, 3 operations maximum on each part, and 3 machine cells arrangement for 2 planning periods, the length of chromosome is 30, the number of rows is 2, and the value of genes is ranged from 1 to 3. This method of developing the chromosome gives an advantage to help the constraint set 4.a (one operation can only be assigned to one cell), since each value of gene in the chromosome can only be one integer which equal to the cell number. Figure 4.8 illustrates this example in the chromosome design used in GA4.

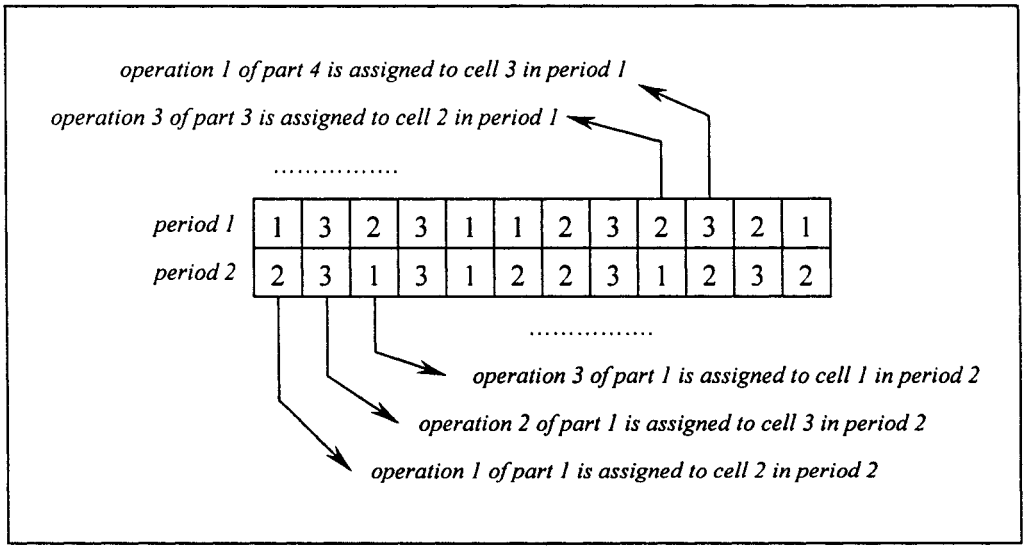

Figure 4.8: Chromosome representation for GA4

The algorithm operates by calling several procedures, which are also the same with GA1.

Main Algorithm (GA4)

1. Clear memory.
2. Input the problem data: parts information, machines information, cells information, processing time of parts on machines, and other information (total time in a period, operator cost, operator idle time cost, lifting risk compensation).
3. Set the GA parameter:
    - population size: default is the largest integer obtained from the number of parts multiplied with the maximum number of operations and divided with 2,
    - maximum number of generation: default is number of cells multiplied with 1000,
    - probability of crossover: default is 0.80,
    - probability of mutation: default is 0.08.
4. Call the Procedure GENERATE, to create string structures for the initial population.
5. Call the Procedure EVALUATION, to evaluate the objective function and fitness value of each string structures, and also calculate the probability of being selected to next generation.
6. Call the Procedure SELECTION, to create the new population by selecting among the eligible records in the old population.
7. Call the Procedure CROSSOVER, to create two new string structures by crossover.
8. Call the Procedure MUTATION, to create new string structure by mutation.
9. Create report for this generation by recording the best objective function and the average objective function value of the population, and also updating the solution data of best objective function.

10. Increase the number of Generation by one.
11. If the Generation = Max Generation then go to step 11, otherwise go to step 5.
12. Create Final Report by finding the best fitness value among the generation record and recording the solution data.
13. Call the Procedure SUMMARY, to create the summary of the best solution.
14. Terminate.

Procedure GENERATE

1. Input the GA parameter, number of parts, number of machines, and number of cells from the Main Algorithm.
2. For each operation on each part in each period, generate random integer number to assign part to cell, range between 1 and number of cells.
3. Increase the number of Population by one.
4. If the Population = Population Size then go to step 5, otherwise go to step 2.
5. Return the value of initial population to the Main Algorithm.
6. Terminate.

Procedure EVALUATION

1. Input the string structures, GA parameter, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration to start evaluation.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.

4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the number of intercellular movements for each part.
7. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.
8. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
9. Calculate the costs: machine cost, machine idle time cost, intercellular movements cost, operator cost, operator idle time cost, and lifting risk cost.
10. Increase the number of Period by one.
11. If the Period = Total Number of Period then go to step 12, otherwise go to step 3.
12. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
13. Calculate the objective function by the summation of all cost components and the penalty.
14. Increase the number of Population by one.
15. If the Population = Population Size then go to step 16, otherwise go to step 3.

16. Calculate the fitness value of each string structure. The fitness values can be calculated using the Equation 4.1.
17. Calculate the probability of being selected to next generation and the cumulative probability.
18. Return the machine cell formation & operator assignment, objective functions, costs & penalties, and string probabilities & cumulative probabilities to the Main Algorithm.
19. Terminate.

Procedure SELECTION

1. Input the string structures, GA parameter, and string probabilities & cumulative probabilities.
2. Create random real number between 0 and 1.
3. Find in which string cumulative probability range that the random number fit, and reproduce the string structure to next generation (also known as the Roulette Wheel Selection).
4. Increase the number of Population by one.
5. If the Population = Population Size then go to step 5, otherwise go to step 2.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure CROSSOVER

1. Input the string structures and GA parameter.

2. For each string structures, create random real number between 0 and 1, and if the random number ≤ probability of crossover then put the string number in Crossover array, otherwise skip.
3. If the length of Crossover array is odd then go back to step 2, otherwise go to step 4.
4. Change the string structures in the Crossover array to pairs of string and put the pairs in the CPair array.
5. For each pair of crossover strings, create random integer number between 1 and 3 to decide the crossover type. Crossover types include 1-cut-point crossover, 2-cut-point crossover, and 3-cut-point crossover.
6. For each pair of crossover string, assign the cut-points based on the crossover type. The cut-points are random integer numbers between 1 and number of parts multiplied with maximum number of operations.
7. For each pair of crossover strings, do crossover to get new generation.
8. Return the new generation of string structures to the Main Algorithm.
9. Terminate.

Procedure MUTATION

1. Input the string structures and GA parameter.
2. For each string structures, create random real number between 0 and 1, and if the random number ≤ probability of mutation then put the string number in Mutation array, otherwise skip.

3. For each mutation strings, create random integer number between 1 and 3 to decide the mutation type. Mutation types include self mutation, switch mutation, and copy mutation.
4. For each mutation string, assign the mutation-points based on the mutation type. The mutation-points are random integer numbers between 1 and number of parts multiplied with maximum number of operations.
5. For each mutation strings, do mutation to get new generation.
6. Return the new generation of string structures to the Main Algorithm.
7. Terminate.

Procedure SUMMARY

1. Input the string structure with best objective function, parts information, machines information, cells information, and other information.
2. Convert the string structures to binary string configuration.
3. Calculate the number of machines needed for each machine type in each cell, and also calculate the machine idle time.
4. Calculate the total number of machines in each cell.
5. Calculate the number of operators needed for each cell, and also calculate the operator idle time.
6. Calculate the number of intercellular movements for each part.
7. Calculate the frequency of lifting for each cell, and make sure the value does not exceed 3 lifts/min.

8. Calculate the composite lifting index for each cell, make sure the value does not exceed 1.5.
9. Calculate the costs: machine cost, machine idle time cost, intercellular movements cost, operator cost, operator idle time cost, and lifting risk cost.
10. Increase the number of Period by one.
11. If the Period = Total Number of Period then go to step 12, otherwise go to step 3.
12. Calculate the penalty for unsatisfied cell capacity (number of machines and number of operators exceeding the capacity) by multiplying the exceeding number of machines with the based penalty, and also by multiplying the exceeding number of operators with the based penalty. The based penalty is obtained from the largest cost components. Penalty also applied for unsatisfied Frequency of Lifting (not exceeding 3 lifts/min) and Composite Lifting Index (not exceeding 1.5) in each cell.
13. Calculate the objective function by the summation of all cost components and the penalty.
14. Return the machine and operator assignment, total machine cost, total machine idle time cost, total operator cost, total operator idle time cost, total lifting risk cost, frequency of lifting in each cell, and composite lifting index in each cell to the Main Algorithm.
15. Terminate.

Final Report

The Final Report from the Genetic Algorithm (GA4) consists of:

1. Part Families (SolF): The string arrangement is mentioned before in Figure 4.8.

2. Machine Groups and Operator Assignments (SolG):

   The string arrangement for SolG is presented in Figure 4.9.

3. Frequency of Lifting (FoL): Array with size Total Period × Total Machine Cells.

4. Composite Lifting Index (CLI): Array with size Total Period × Total Machine Cells.

5. Cost components: Machine Cost (MC), Machine Idle Time Cost (MIC), Operator Cost (WC), Operator Idle Time Cost (WIC), Intercellular Movements Cost (IMC), Lifting Risk Cost (LC), Machine Relocation Cost (MCCost), and Manpower Level Change Cost (WCCost).

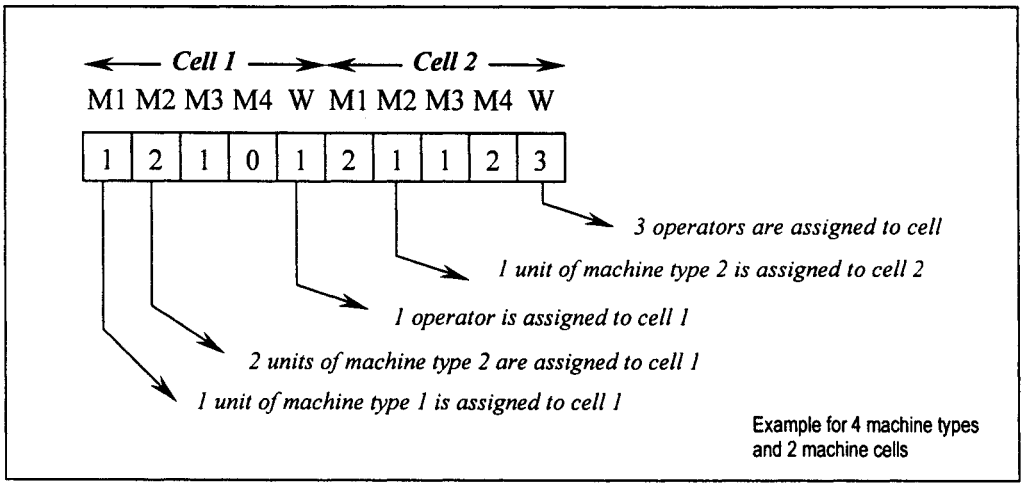

Figure 4.9: Machine Groups and Operator Assignments array

Chapter 5 – Numerical Examples

In this chapter the mathematical models developed under single and multi period planning horizon are tested by using several numerical examples of different sizes. The examples are solved using three methods:

1. Sequentially using combination of Branch-and-Bound and Heuristic (SeqBBH), which consists of two steps: Step 1, solving the group technology problem (part family and machine grouping), and Step 2 using the result from the first model to solve the manual lifting tasks problem (operator assignment).
2. Simultaneously using combination of Branch-and-Bound and Heuristic (SimBBH), which is solving the models concurrently.
3. Simultaneously using Genetic Algorithms (SimGA), which are developed to deal with nonlinearity in the models and to solve large-scale problems.

The first method and the second method are developed using the LINGO version 7.0 by Lindo Systems. The source codes are attached in Appendix I. The third method is developed using MATLAB version 6.5.0.180913a release 13 by MathWorks. The source codes are attached in Appendix II. The models are run on IBM compatible PC with AMD 1.3 GHz processor and 256 megabytes random access memory.

Furthermore, the results from all three methods are compared and analyzed in terms of minimizing the objective function and time consumption.

5.1. Model 1: Single Period

In this section, Model 1 is tested using the three methods and each method with two numerical examples of different sizes. Some input data for the examples are taken from Ozdemir (1995). The first example considers 8 different part types, 5 different machine types, and 2 cells are to be formed. The second example considers 15 different part types, 10 different machine types, and 3 cells are to be formed.

5.1.1. Example 1: Model 1 for Two-Cell Formation

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is $10 per hour, operator idle time cost is $4 per hour, and the approximate lifting risk compensation is $15,000 per operator per lifting index. The cell capacity for cell 1 is 8 machines, and for cell 2 is 7 machines. The maximum number of operators for each cell is 6.

Machine requirements, annual demand, load weight, and the vertical manual lifting distance for each part are given in Table 5.1. The machine information is given in Table 5.2.

Table 5.1: Machine requirements, annual demand, load weight, and vertical lifting distance

Part Type	Machining Time *(second)*					Demand *(unit)*	Load Weight *(kg)*	Vertical Lifting Distance *(cm)*
	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5			
1	120	240	0	180	0	22000	9	40
2	0	192	0	108	0	24000	17	45
3	0	0	132	0	210	18000	15	50
4	0	120	0	138	0	17500	16	60
5	180	156	0	0	0	20000	11	40
6	0	114	126	0	144	21000	10	45
7	0	0	150	0	162	16000	12	55
8	156	174	0	144	0	19000	8	35

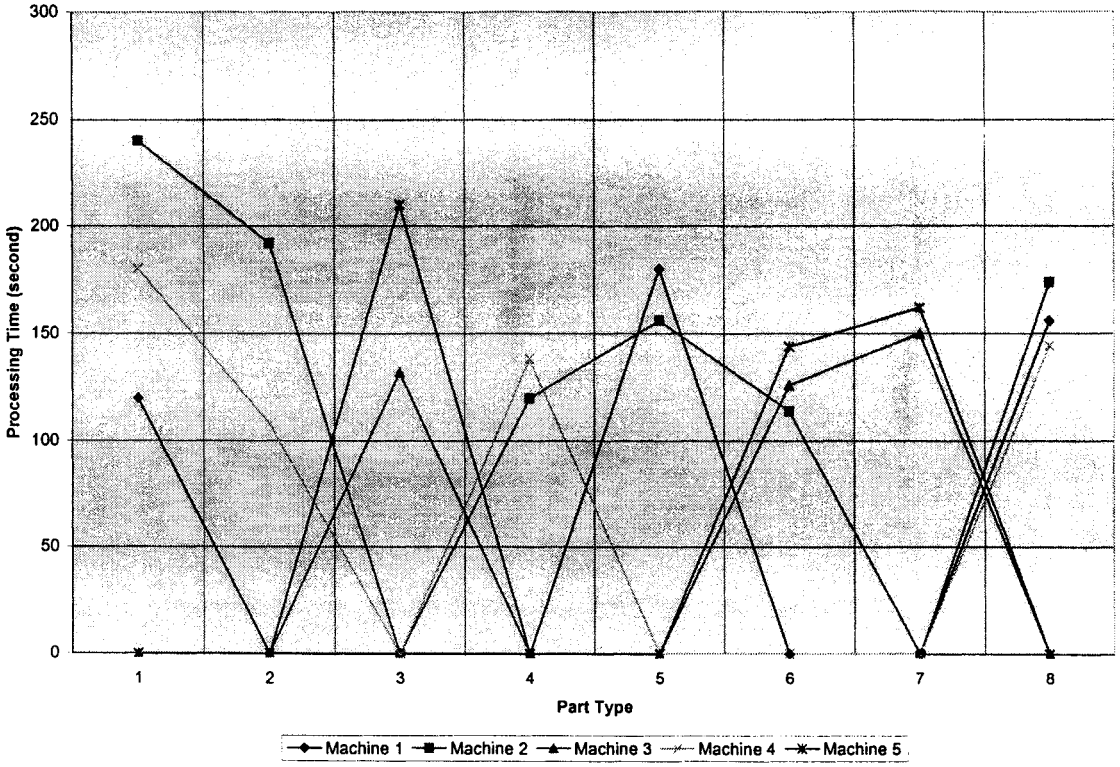

Figure 5.1: Processing time of each part

Table 5.2: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	16000	0.75	4
2	20000	0.6	6
3	18000	0.8	5
4	12500	0.7	6
5	14000	0.65	3

5.1.1.1. Result of SeqBBH Method

Example 1 is solved sequentially: Step 1, part families and machine cells formations are solved in 1 second with global optimal state; and Step 2, operator assignment is solved in 2 seconds with local optimal state.

The first cell contains: 2 units of each machine type 1 and 2; and 1 unit of each machine type 3, 4, and 5. The second cell contains: 1 unit of each machine type 2, 3, 4, and 5.

For part families, the first part family consists of part type 1, 3, 5, 6, and 8, whereas the second part family consists of part type 2, 4, and 7. Because the model does not allow intercellular movements, no part is assigned to more than one cell.

For the first cell, the number of operators assigned is 4, the average frequency of lifting is 0.546 lifts per minute, and the composite lifting index is 1.175. For the second cell, the number of operators assigned is 2, the average lifting frequency is 0.479 lifts per minute, and the composite lifting index is 1.398.

The results of part families and machine cells formations are summarized in Table 5.3. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.4. Also, various cost results of the solution are given in Table 5.5.

Table 5.3: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*					
		1	2	3	4	5	Total
Cell 1	1, 3, 5, 6, 8	2	2	1	1	1	7
Cell 2	2, 4, 7	0	1	1	1	1	4

Table 5.4: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	4	0.546	1.175
Cell 2	2	0.479	1.398

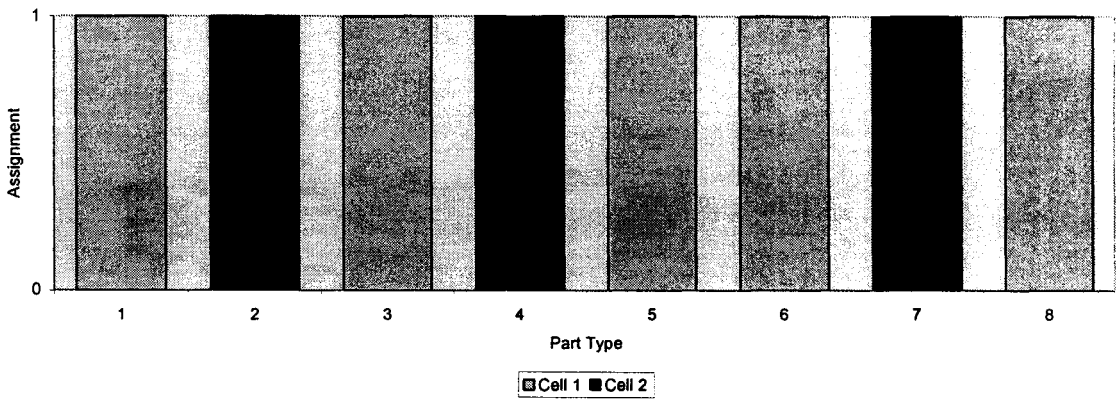

Figure 5.2: Part families

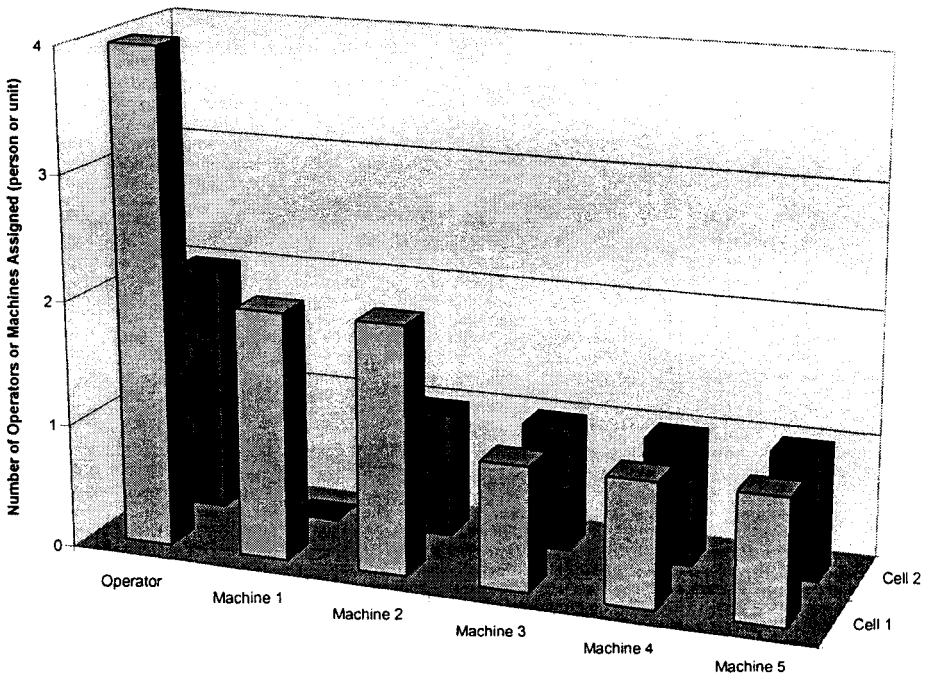

Figure 5.3: Operator and machine assignment

Table 5.5: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	181,000.00
Machine Idle Time Cost	25,450.00
Operator Cost	120,000.00
Operator Idle Time Cost	3,972.33
Lifting Risk Cost	112,406.00
Total Cost	**442,828.33**

5.1.1.2. Result of SimBBH Method

Example 1 is solved simultaneously for the part families, machine cells formations, and operator assignment. This example is solved in 4 minutes and 59 seconds with local optimal state.

The first cell contains: 2 units of each machine type 1 and 2; and 1 unit of machine type 3, 4, and 5; and the second cell contains 1 unit of each machine type 2, 3, 4, and 5.

For part families, the first cell family consists of part type 1, 5, 6, 7, and 8, whereas the second cell family consists of part 2, 3, and 4. Because the model does not allow intercellular movements, no part is assigned to more than one cell.

For the first cell, the number of operators assigned is 4, the average frequency of lifting is 0.538 lifts per minute, and the composite lifting index is 0.978. For the second cell, the number of operators assigned is 2, the average lifting frequency is 0.496 lifts per minute, and the composite lifting index is 1.422.

The results of part families and machine cells formations are summarized in Table 5.6. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.7. Also, various cost results of the solution are given in Table 5.8.

Table 5.6: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*					
		1	2	3	4	5	Total
Cell 1	1, 5, 6, 7, 8	2	2	1	1	1	7
Cell 2	2, 3, 4	0	1	1	1	1	4

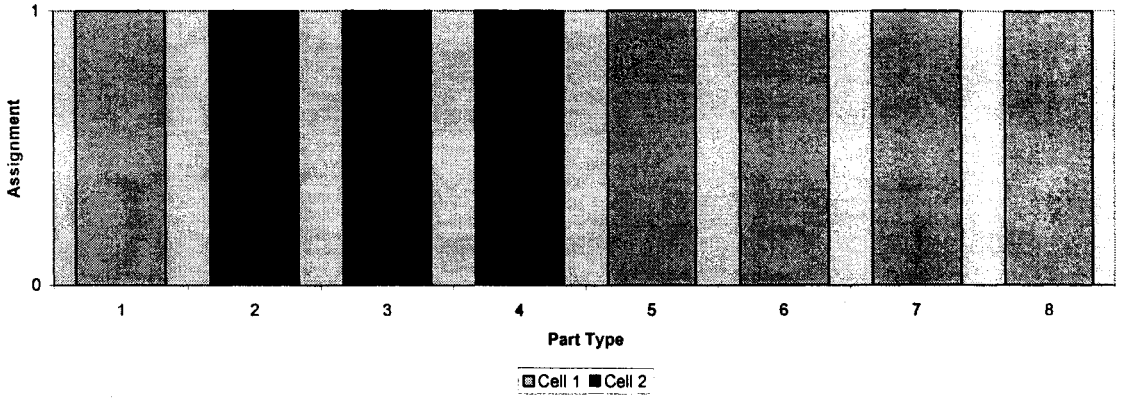

Figure 5.4: Part families

Table 5.7: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	4	0.538	0.978
Cell 2	2	0.496	1.422

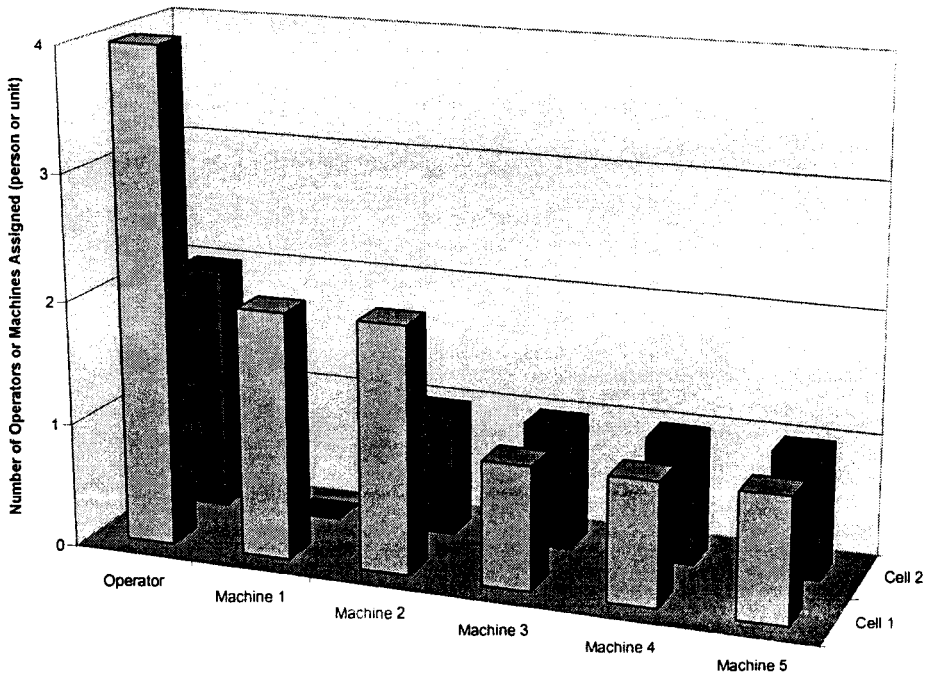

Figure 5.5: Operator and machine assignment

Table 5.8: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	181,000.00
Machine Idle Time Cost	25,450.00
Operator Cost	120,000.00
Operator Idle Time Cost	3,972.33
Lifting Risk Cost	101,349.40
Total Cost	**431,771.73**

5.1.1.3. Result of SimGA Method (GA1)

Example 1 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA1. The example is run for 3000 generations, with the population of 10 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 493rd generation and the total time consumed is 52 seconds (see Figure 5.6).

The first cell contains 1 unit of each machine type 2, 3, 4, and 5. The second cell contains: 2 units of each machine type 1 and 2; and 1 unit of each machine type 3, 4, and 5.

For part families, the first part family consists of part type 2, 3, and 4; whereas the second part family consists of part type 1, 5, 6, 7, and 8. Because the model does not allow intercellular movements, no part is assigned to more than one cell.

For the first cell, the number of operators assigned is 2, the average frequency of lifting is 0.496 lifts per minute, and the composite lifting index is 1.422. For the second cell, the number of operators assigned is 4, the average lifting frequency is 0.538 lifts per minute, and the composite lifting index is 0.978.

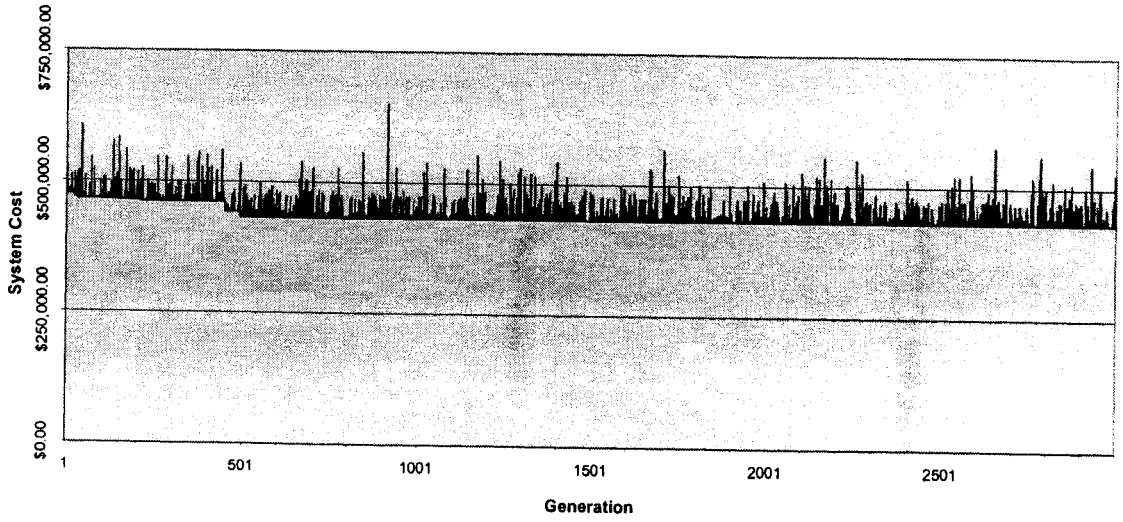

Figure 5.6: Average population cost and best cost for each generation

The results of part families and machine cells formations are summarized in Table 5.9. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.10. Also, various cost results of the solution are given in Table 5.11.

Table 5.9: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*					
		1	2	3	4	5	Total
Cell 1	2, 3, 4	0	1	1	1	1	4
Cell 2	1, 5, 6, 7, 8	2	2	1	1	1	7

Table 5.10: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	2	0.496	1.422
Cell 2	4	0.538	0.978

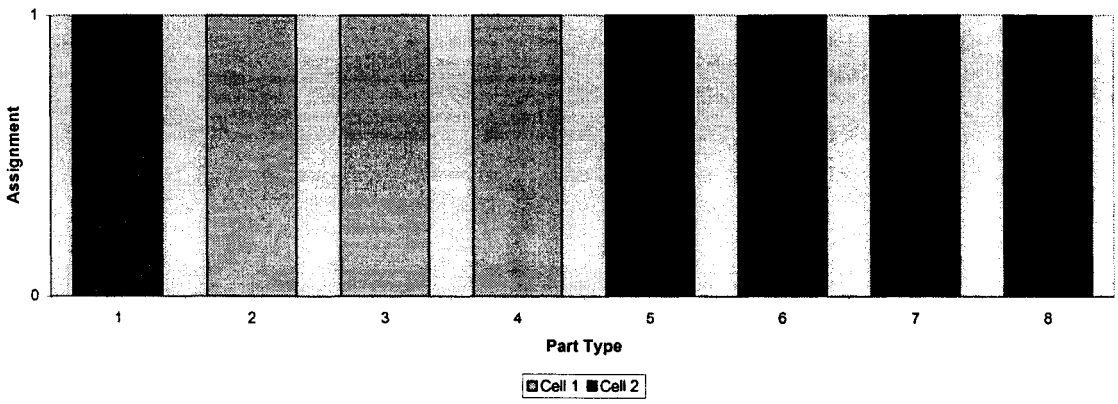

Figure 5.7: Part families

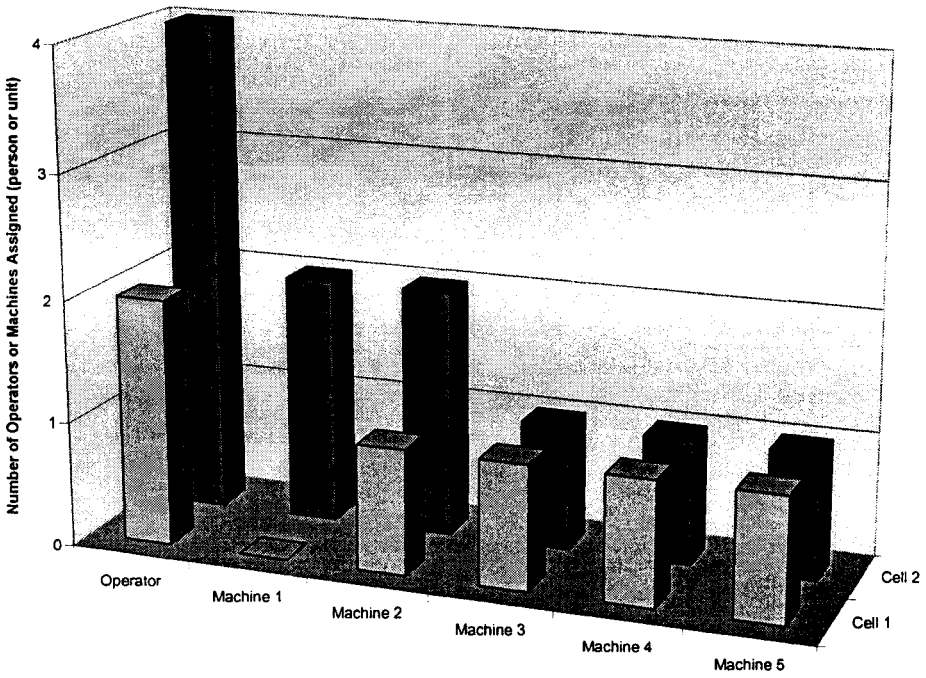

Figure 5.8: Operator and machine assignment

Table 5.11: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	181,000.00
Machine Idle Time Cost	25,450.00
Operator Cost	120,000.00
Operator Idle Time Cost	3,972.33
Lifting Risk Cost	101,349.40
Total Cost	**431,771.73**

5.1.1.4. Comparison of Results of Example 1

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 1 has:

- 85 total variables, including 28 integer variables;
- 96 total constraints, including 39 nonlinear constraints; and
- 342 total nonzeros, including 134 nonlinear nonzeros.

The SeqBBH method has the fastest running time, but the objective function value is not the best among the three. The SimBBH method has the slowest running time, but the system cost resulted is the best among the three. The SimGA method solved the problem in a reasonable time and also gave the best system cost among the three.

So, if SimGA method compared with SimBBH method, the reduction in computational time is 82.61% (299 seconds to 52 seconds), and the system costs are the same. The detailed comparison of the three methods is given in Table 5.12 and Figure 5.9.

Table 5.12: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: Local Optimal	Local Optimal	Best Solution found after 493rd generation
Solver Time	00:00:03	00:04:59	00:00:52
Cost Elements:			
Machine Capital Cost	*181,000.00*	*181,000.00*	*181,000.00*
Machine Idle Time Cost	*25,450.00*	*25,450.00*	*25,450.00*
Operator Cost	*120,000.00*	*120,000.00*	*120,000.00*
Operator Idle Time Cost	*3,972.33*	*3,972.33*	*3,972.33*
Lifting Risk Cost	*112,406.00*	*101,349.40*	*101,349.40*
Total System Cost	442,828.33	431,771.73	431,771.73

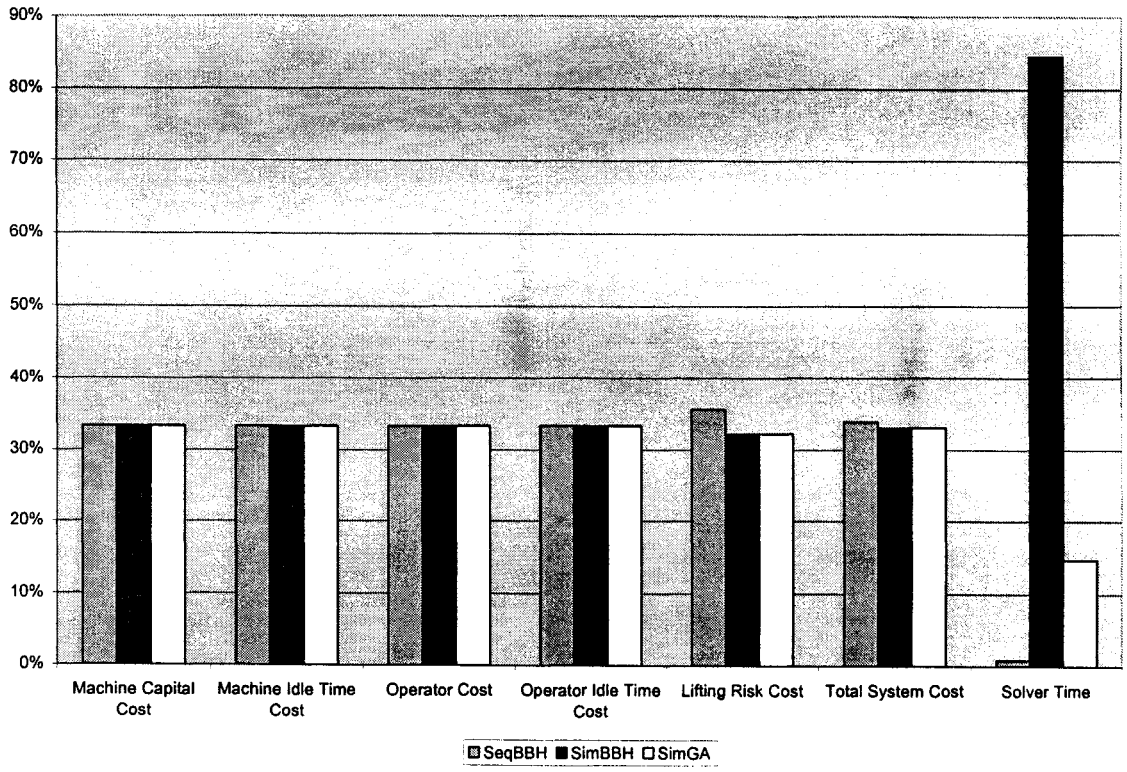

Figure 5.9: Costs and time comparisons

5.1.2. Example 2: Model 1 for Three-Cell Formation

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is $10 per hour, operator idle time cost is $4 per hour, and the approximate lifting risk compensation is $15,000 per operator per lifting index. The cell capacity for cell 1 is 7 machines, for cell 2 is 6 machines, and for cell 3 is 8 machines. The maximum number of operators allowed for cell 1 and cell 2 is 5, and for cell 3 is 4.

The annual demand, load weight, and vertical manual lifting distance for each part are given in Table 5.13. The machine requirements and the machine information are given in Table 5.14 and Table 5.15, respectively.

Table 5.13: Annual demand, load weight, and vertical lifting distance

Part Type	Demand *(unit)*	Load Weight *(kg)*	Vertical Lifting Distance *(cm)*
1	21000	17	40
2	15000	13	45
3	17000	7	45
4	18000	10	47
5	16000	9	52
6	17000	11	38
7	16500	7	48
8	16000	15	50
9	20000	6	60
10	18500	11	40
11	16000	12	45
12	18000	10	55
13	19000	8	35
14	17500	9	50
15	17000	12	40

Table 5.14: Machine requirements

Part Type	Machining Time *(second)*									
	M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10
1	0	0	108	114	0	132	0	0	0	0
2	0	0	0	0	144	0	120	0	0	0
3	0	0	0	0	0	0	0	138	0	120
4	0	0	0	0	84	0	132	120	0	96
5	0	192	0	186	0	0	0	0	144	0
6	156	144	0	0	0	0	0	0	120	0
7	0	0	0	180	120	0	120	0	0	0
8	0	0	102	108	0	168	0	0	0	0
9	168	78	0	0	48	0	0	0	0	0
10	0	0	0	0	108	0	108	0	0	0
11	0	0	0	0	114	0	132	126	0	90
12	0	0	0	192	0	0	102	0	0	0
13	0	0	174	0	0	0	0	0	108	0
14	0	0	0	0	120	0	0	0	0	90
15	120	0	0	0	126	0	96	0	0	0

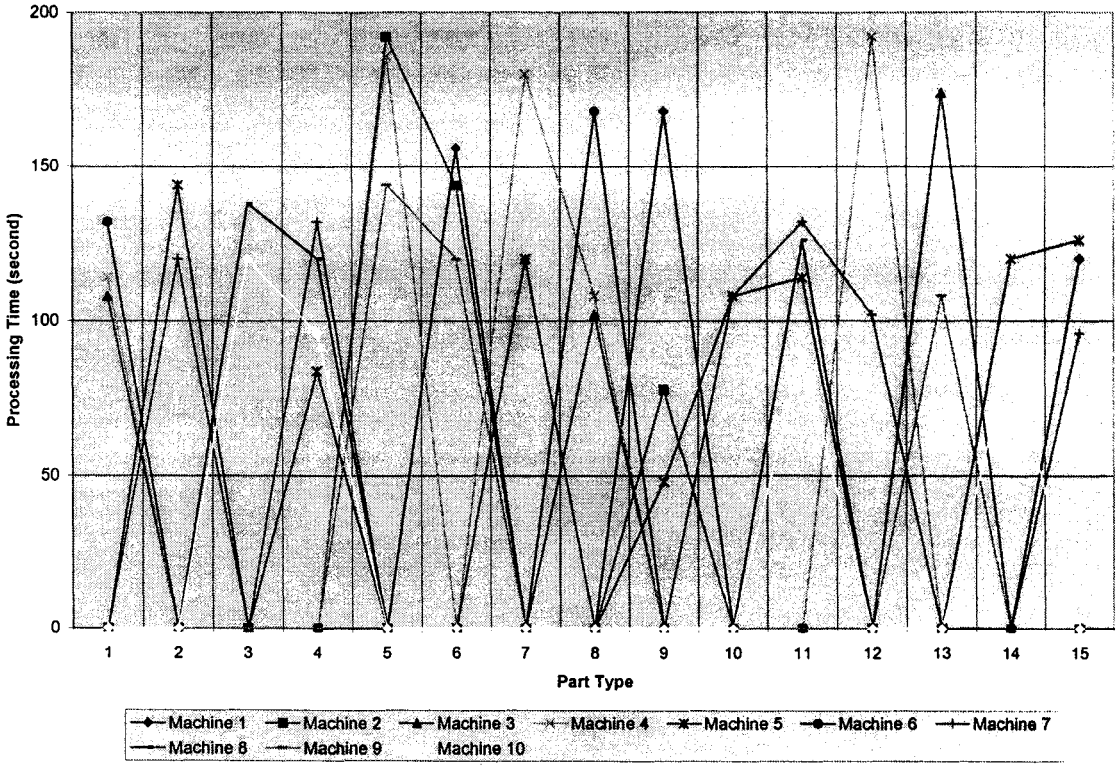

Figure 5.10: Processing time of each part

Table 5.15: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	25000	0.75	4
2	26000	0.5	5
3	24000	0.45	3
4	27000	0.6	5
5	28000	0.5	4
6	29000	0.6	6
7	23000	0.8	5
8	25000	0.7	6
9	30000	0.65	3
10	28000	0.55	4

5.1.2.1. Result of SeqBBH Method

Example 2 is solved sequentially: Step 1, part families and machine cells formations are solved in 14 minutes and 5 seconds with global optimal state; and Step 2, operator assignment is solved in 4 seconds with local optimal state.

The first cell contains 1 unit of each machine type 1, 2, 3, 4, 5, 7, and 9. The second cell contains 1 unit of each machine type 5, 7, 8, and 10. The third cell contains 1 unit of each machine type 1, 3, 4, 5, 6, and 7.

For part families, the first part family consists of part type 2, 5, 6, 9, 10, 12, and 13. The second part family consists of part type 3, 4, 11, and 14. The third part family consists of part type 1, 7, 8, and 15. Because the model does not allow intercellular movements, no part is assigned to more than one cell.

For the first cell, the number of operators assigned is 4, the average frequency of lifting is 0.625 lifts per minute, and the composite lifting index is 1.024. For the second cell, the number of operators assigned is 3, the average lifting frequency is 0.569 lifts per minute, and the composite lifting index is 0.961. For the third cell, the number of operators assigned is 3, the average lifting frequency is 0.588 lifts per minute, and the composite lifting index is 1.328.

The results of part families and machine cells formations are summarized in Table 5.16. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.17. Also, various cost results of the solution are given in Table 5.18.

Table 5.16: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*										
		1	2	3	4	5	6	7	8	9	10	Total
Cell 1	2, 5, 6, 9, 10, 12, 13	1	1	1	1	1	0	1	0	1	0	7
Cell 2	3, 4, 11, 14	0	0	0	0	1	0	1	1	0	1	4
Cell 3	1, 7, 8, 15	1	0	1	1	1	1	1	0	0	0	6

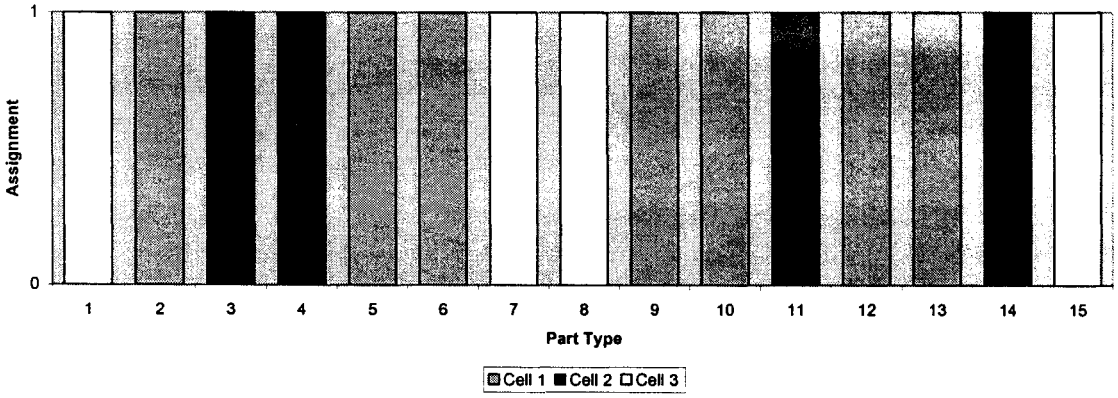

Figure 5.11: Part families

Table 5.17: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	4	0.625	1.024
Cell 2	3	0.569	0.961
Cell 3	3	0.588	1.328

Table 5.18: Final cost values

Cost Function	Cost Value *($)*
Machine Capital Cost	443,000.00
Machine Idle Time Cost	38,213.33
Operator Cost	200,000.00
Operator Idle Time Cost	18,899.17
Lifting Risk Cost	164,457.40
Total Cost	**864,569.90**

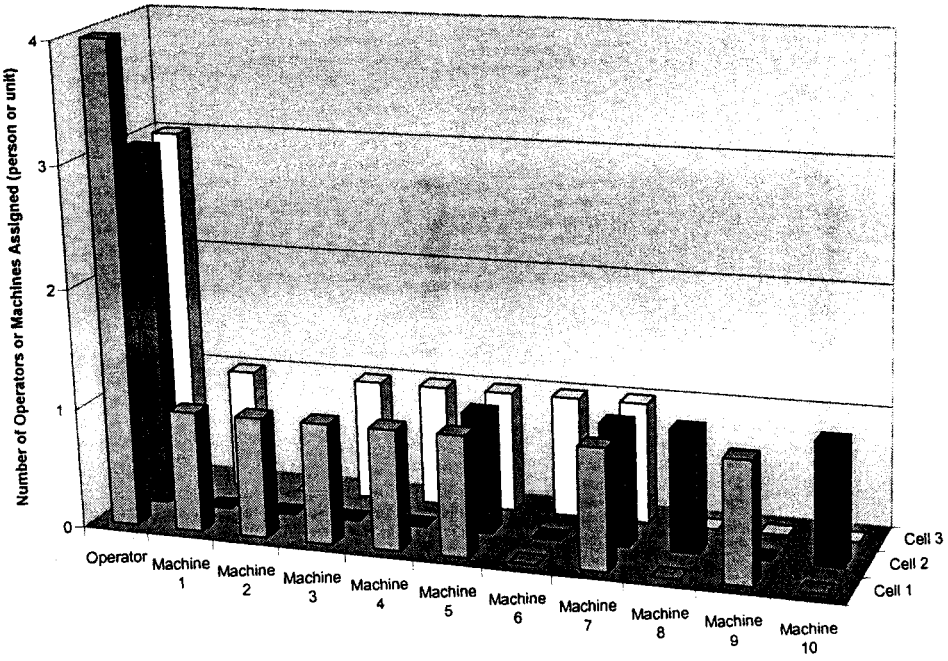

Figure 5.12: Operator and machine assignment

5.1.2.2. Result of SimBBH Method

Trials have been made to solve Example 2 simultaneously for the part families, machine cells formations, and operator assignment. However, after 10 minutes and 20 seconds, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.1.2.3. Result of SimGA Method (GA1)

Example 2 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA1. The example is run for 4500 generations, with the population of 15 chromosomes, probability of crossover (p_c) equal

to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 4367th generation and the total time consumed is 5 minutes 15 seconds (see Figure 5.13).

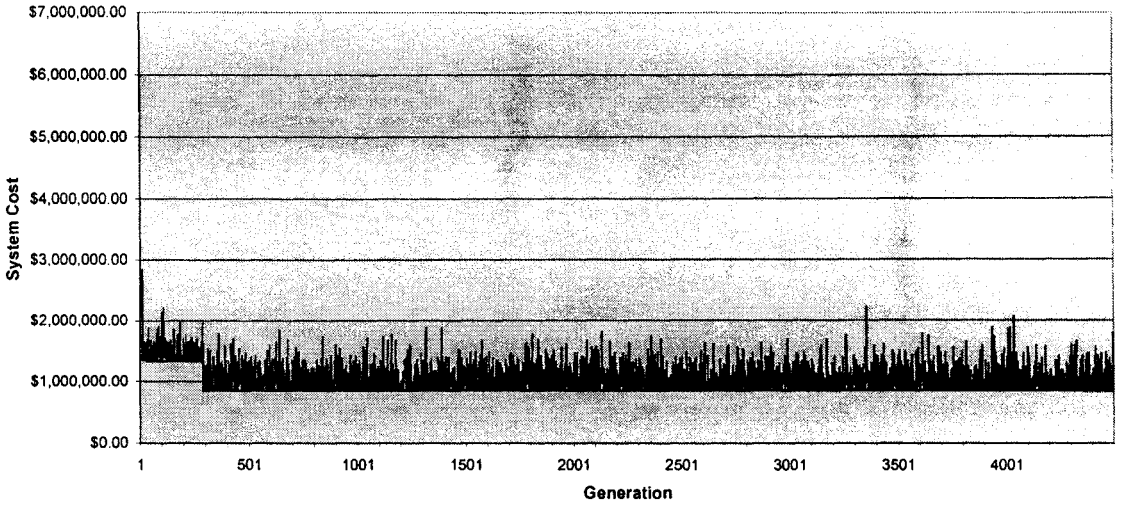

Figure 5.13: Average population cost and best cost for each generation

The first cell contains 1 unit of each machine type 3, 4, 5, 7, 8, 9, and 10. The second cell contains 1 unit of each machine type 1, 2, 4, 5, 7, and 9. The third cell contains 1 unit of each machine type 1, 3, 4, 5, 6, and 7.

For part families, the first part family consists of part type 3, 4, 11, 12, 13, and 14. The second part family consists of part type 5, 6, 7, 9, and 10. The third part family consists of part type 1, 2, 8, and 15. Because the model does not allow intercellular movements, no part is assigned to more than one cell.

For the first cell, the number of operators assigned is 3, the average frequency of lifting is 0.775 lifts per minute, and the composite lifting index is 0.962. For the second cell, the number of operators assigned is 3, the average lifting frequency is 0.682 lifts per minute, and the composite lifting index is 0.870. For the third cell, the number of

operators assigned is 2, the average lifting frequency is 0.8 lifts per minute, and the composite lifting index is 1.374.

The results of part families and machine cells formations are summarized in Table 5.19. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.20. Also, various cost results of the solution are given in Table 5.21.

Table 5.19: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*										
		1	2	3	4	5	6	7	8	9	10	Total
Cell 1	3, 4, 11, 12, 13, 14	0	0	1	1	1	0	1	1	1	1	7
Cell 2	5, 6, 7, 9, 10	1	1	0	1	1	0	1	0	1	0	6
Cell 3	1, 2, 8, 15	1	0	1	1	1	1	1	0	0	0	6

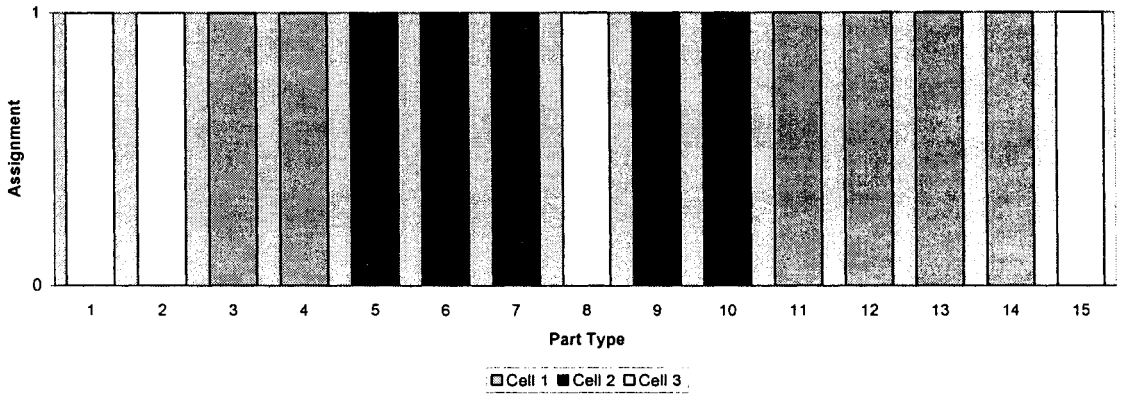

Figure 5.14: Part families

Table 5.20: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	3	0.775	0.962
Cell 2	3	0.682	0.870
Cell 3	2	0.800	1.374

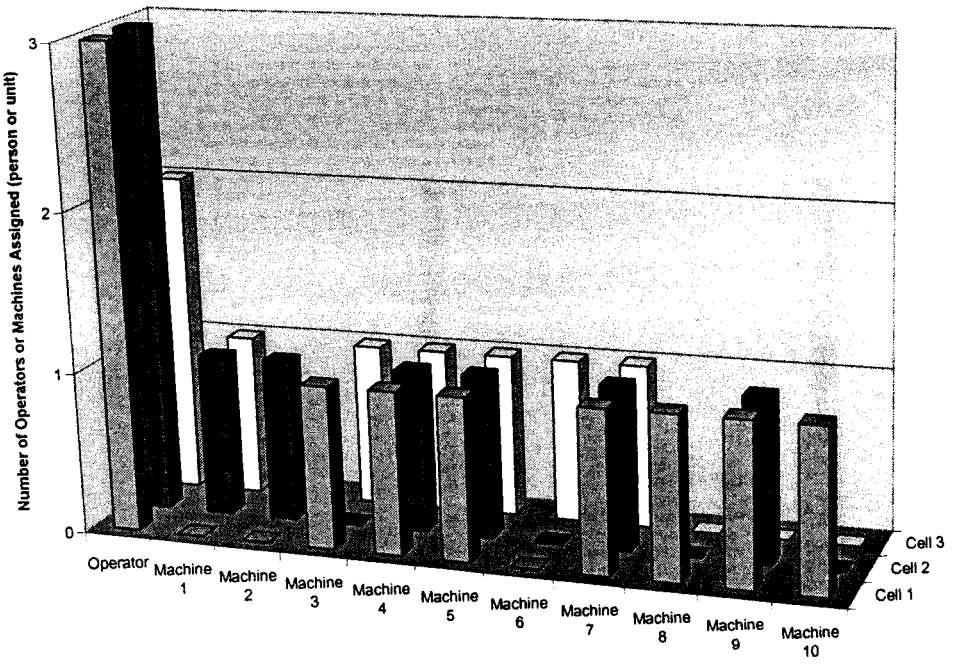

Figure 5.15: Operator and machine assignment

Table 5.21: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	500,000.00
Machine Idle Time Cost	54,213.33
Operator Cost	160,000.00
Operator Idle Time Cost	2,899.17
Lifting Risk Cost	123,655.20
Total Cost	**840,767.70**

5.1.2.4. Comparison of Results of Example 2

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 2 has:

- 215 total variables, including 78 integer variables;

- 231 total constraints, including 100 nonlinear constraints; and
- 931 total nonzeros, including 369 nonlinear nonzeros.

The SeqBBH method is not the best in terms of both running time and objective function value. The SimBBH method cannot solve this example. This situation happens because the nonlinearity in the model makes the solver stuck in infeasibility state. The SimGA method solved the problem with the least running time and also gave the best system cost.

So, if SimGA method compared with SeqBBH method, the reduction in computational time is 62.90% (849 seconds to 315 seconds), and the reduction in system costs is 2.75% ($864,569.90 to $840,767.70).

The detailed comparison of the three methods is given in Table 5.22.

Table 5.22: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: Local Optimal	No Feasible Solution	Best Solution found after 4367th generation
Solver Time	00:14:09	00:10:20	00:05:15
Cost Elements:			
Machine Capital Cost	*443,000.00*	-	*500,000.00*
Machine Idle Time Cost	*38,213.33*	-	*54,213.33*
Operator Cost	*200,000.00*	-	*160,000.00*
Operator Idle Time Cost	*18,899.17*	-	*2,899.17*
Lifting Risk Cost	*164,457.40*	-	*123,655.20*
Total System Cost	864,569.90	-	840,767.70

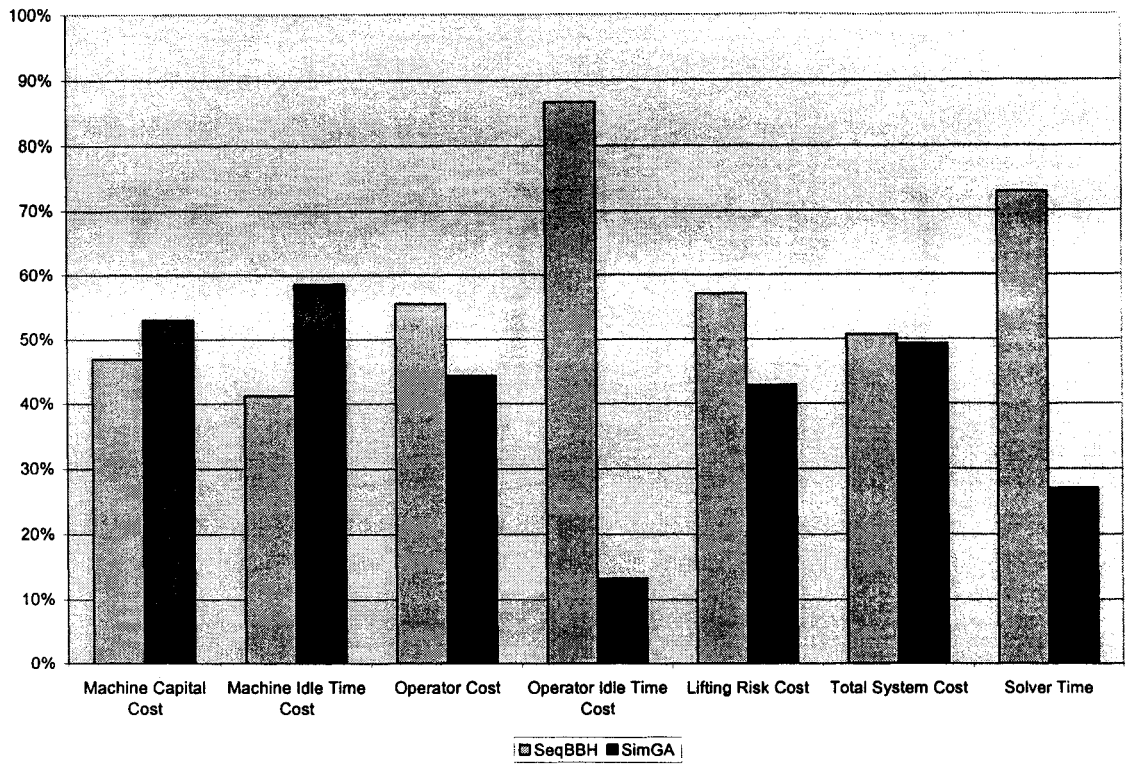

Figure 5.16: Costs and time comparisons

5.2. Model 2: Single Period with Job Sequence

In this section, Model 2 is tested using the three methods and each method with two numerical examples of different sizes. Some input data for the examples are taken from Ozdemir (1995). The first example considers 9 different part types, each part consists of 4 operations or less, 6 different machine types, and 2 cells are to be formed. The second example considers 15 different part types, 10 different machine types, and 3 cells are to be formed.

5.2.1. Example 3: Model 2 for Two-Cell Formation

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is $10 per hour, operator idle time cost is $4 per hour, and the approximate lifting risk compensation is $15,000 per operator per lifting index. The cell capacity for cell 1 is 8 machines, and for cell 2 is 10 machines. The maximum number of operators allowed for cell 1 is 6 and for cell 2 is 7.

The annual demand, load weight, vertical manual lifting distance, and intercellular movement cost for each part are given in Table 5.23. The machine requirements and the machine information are given in Table 5.24 and Table 5.25, respectively.

Table 5.23: Annual demand, load weight, and vertical lifting distance

Part Type	Demand *(unit)*	Load Weight *(kg)*	Vertical Lifting Distance *(cm)*	Intercellular Movement Cost *($)*
1	22000	10	45	0.3
2	24000	9	50	0.5
3	30000	17	41	0.4
4	27000	14	60	0.35
5	20000	12	67	0.25
6	21000	6	45	0.4
7	24000	8	35	0.3
8	19000	20	40	0.35
9	27000	5	40	0.4

Table 5.24: Machine requirements

Part Type & Jobs		Machining Time *(second)*					
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6
Part 1	Job A	120	0	0	0	0	0
	Job B	0	240	0	0	0	0
	Job C	0	0	0	180	0	0
	Job D	0	0	0	0	0	0
Part 2	Job A	0	0	0	0	0	102
	Job B	0	0	0	0	108	0
	Job C	0	192	0	0	0	0
	Job D	0	0	0	0	0	0
Part 3	Job A	0	0	132	0	0	0
	Job B	0	0	0	0	210	0
	Job C	0	0	0	0	0	0
	Job D	0	0	0	0	0	0
Part 4	Job A	0	0	0	138	0	0
	Job B	0	120	0	0	0	0
	Job C	0	0	0	0	0	180
	Job D	0	0	0	0	0	0
Part 5	Job A	0	156	0	0	0	0
	Job B	0	0	0	0	0	168
	Job C	0	0	0	0	0	0
	Job D	0	0	0	0	0	0
Part 6	Job A	0	0	0	0	144	0
	Job B	0	0	126	0	0	0
	Job C	0	114	0	0	0	0
	Job D	0	0	0	0	0	0
Part 7	Job A	0	174	0	0	0	0
	Job B	156	0	0	0	0	0
	Job C	0	0	0	144	0	0
	Job D	0	0	0	0	0	132
Part 8	Job A	0	0	0	0	132	0
	Job B	0	0	120	0	0	0
	Job C	0	0	0	0	0	120
	Job D	0	0	0	0	0	0
Part 9	Job A	0	0	168	0	0	0
	Job B	0	192	0	0	0	0
	Job C	0	0	0	120	0	0
	Job D	180	0	0	0	0	0

Table 5.25: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	16000	0.75	4
2	20000	0.6	6
3	18000	0.8	5
4	12500	0.7	6
5	14000	0.65	3
6	17000	0.55	4

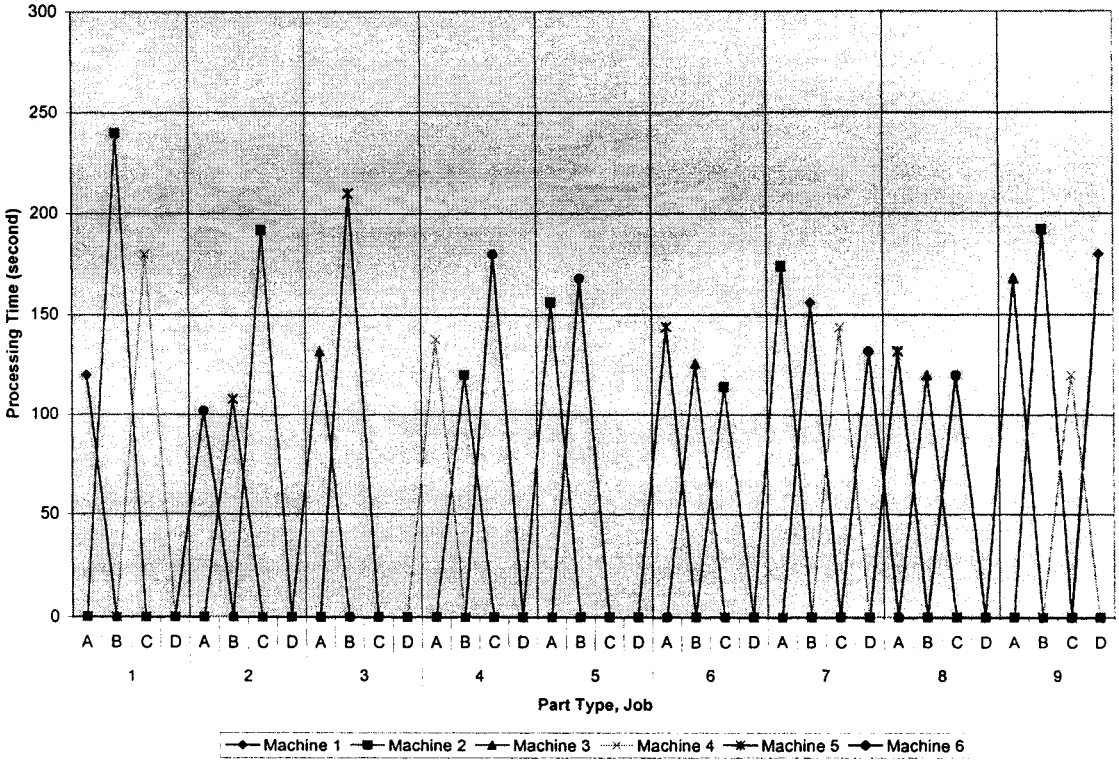

Figure 5.17: Processing time of each job on each part

5.2.1.1. Result of SeqBBH Method

Example 3 is solved sequentially: Step 1, part families and machine cells formations are solved in 1 minute and 49 seconds with global optimal state; and Step 2, operator assignment is solved in 3 minutes and 52 seconds with local optimal state.

The first cell contains: 1 unit of each machine type 2, 3, and 6; and 3 units of machine type 5. The second cell contains: 2 units of each machine type 1, 4, and 6; 3 unit of machine type 2; and 1 unit of machine type 3.

For part families, the first part family consists of part type 2, 3, 6, and 8. The second part family consists of part type 1, 4, 5, 7, 8, and 9. Intercellular movements occur on part type 8.

For the first cell, the number of operators assigned is 5, the average frequency of lifting is 0.388 lifts per minute, and the composite lifting index is 1.500. For the second cell, the number of operators assigned is 6, the average lifting frequency is 0.569 lifts per minute, and the composite lifting index is 1.471.

The results of part families and machine cells formations are summarized in Table 5.26. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.27. Also, various cost results of the solution are given in Table 5.28.

Table 5.26: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*						
		1	2	3	4	5	6	Total
Cell 1	2, 3, 6, 8A, 8C, 8D	0	1	1	0	3	1	6
Cell 2	1, 4, 5, 7, 8B, 9	2	3	1	2	0	2	10

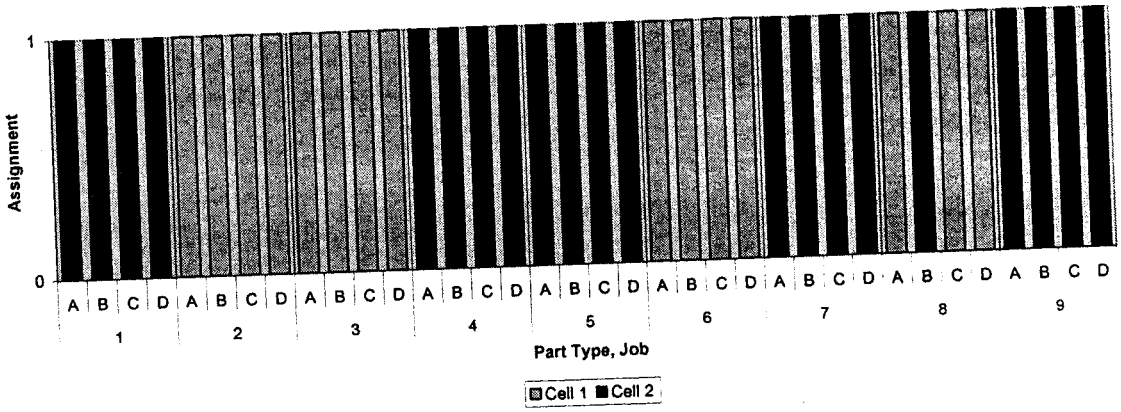

Figure 5.18: Part families

Table 5.27: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	5	0.388	1.500
Cell 2	6	0.569	1.471

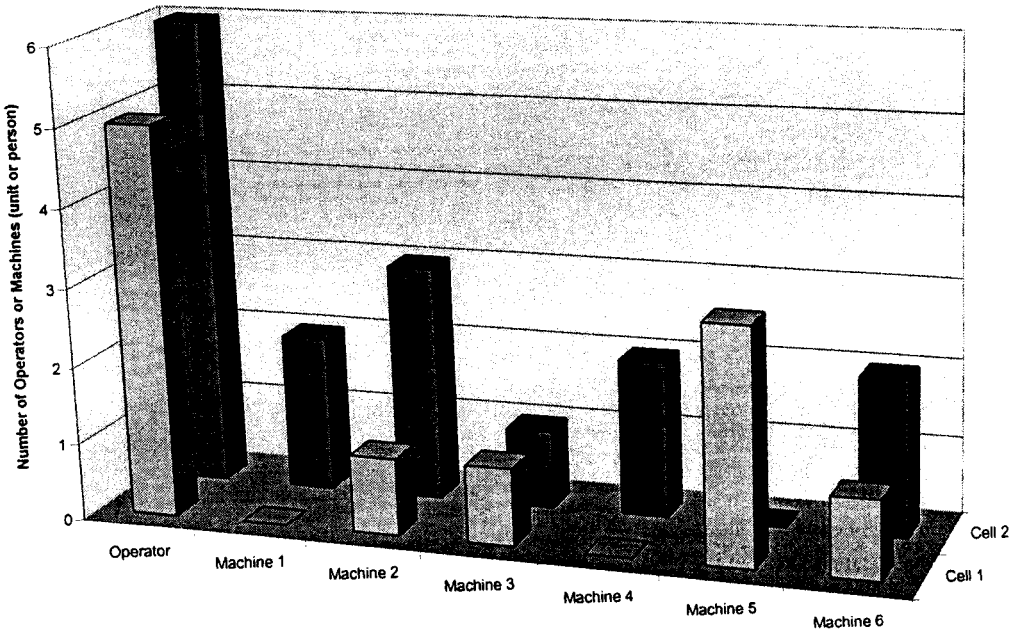

Figure 5.19: Operator and machine assignment

Table 5.28: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	266,000.00
Machine Idle Time Cost	18,298.33
Intercellular Movements Cost	13,300.00
Operator Cost	220,000.00
Operator Idle Time Cost	16,579.33
Lifting Risk Cost	244,924.80
Total Cost	**779,102.46**

5.2.1.2. Result of SimBBH Method

Trials have been made to solve Example 3 simultaneously for the part families, machine cells formations, and operator assignment. However, after 6 minutes and 48 seconds, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.2.1.3. Result of SimGA Method (GA2)

Example 3 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA2. The example is run for 3000 generations, with the population of 18 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution obtained after 687^{th} generation and the total time consumed is 2 minutes 28 seconds (see Figure 5.20).

The first cell contains: 2 units of each machine type 2 and 6; and 1 unit of each machine type 3, 4, and 5. The second cell contains: 2 units of each machine type 1, 2, and 5; and 1 unit of each machine type 3, 4, and 6.

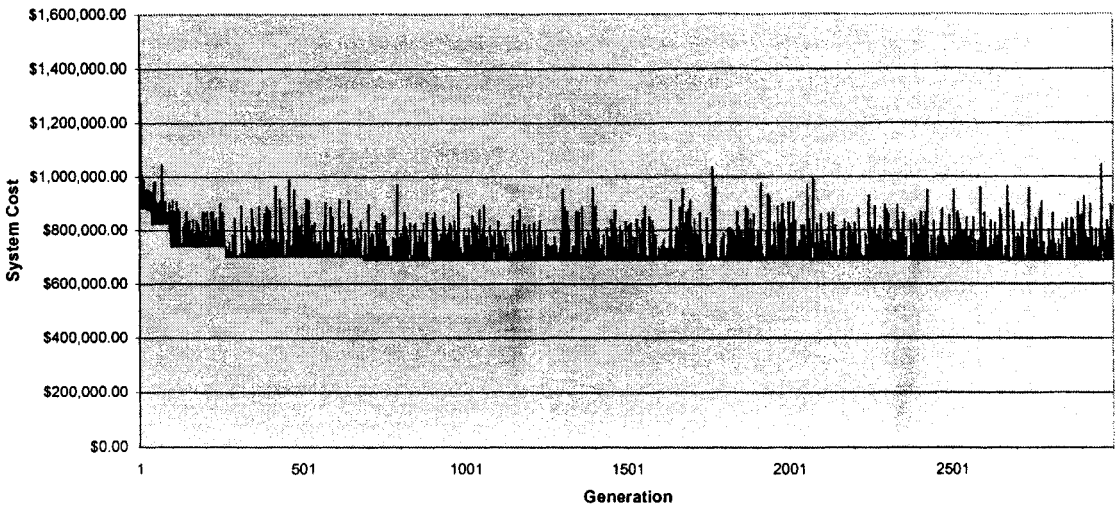

Figure 5.20: Average population cost and best cost for each generation

For part families, the first part family consists of part type 4, 5, 6, 7, 8, and 9. The second part family consists of part type 1, 2, 3, 7, 8, and 9. Intercellular movements occur on part type 7, 8, and 9.

For the first cell, the number of operators assigned is 4, the average frequency of lifting is 0.635 lifts per minute, and the composite lifting index is 1.473. For the second cell, the number of operators assigned is 5, the average lifting frequency is 0.563 lifts per minute, and the composite lifting index is 1.492.

The results of part families and machine cells formations are summarized in Table 5.29. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.30. Also, various cost results of the solution are given in Table 5.31.

Table 5.29: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*						
		1	2	3	4	5	6	Total
Cell 1	4, 5, 6, 7C, 7D, 8C, 8D, 9A, 9B	0	2	1	1	1	2	7
Cell 2	1, 2, 3, 7A, 7B, 8A, 8B, 9C, 9D	2	2	1	1	2	1	9

Table 5.30: Operator assignment, average frequency of lifting, and the composite lifting index

	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Cell 1	4	0.635	1.473
Cell 2	5	0.563	1.492

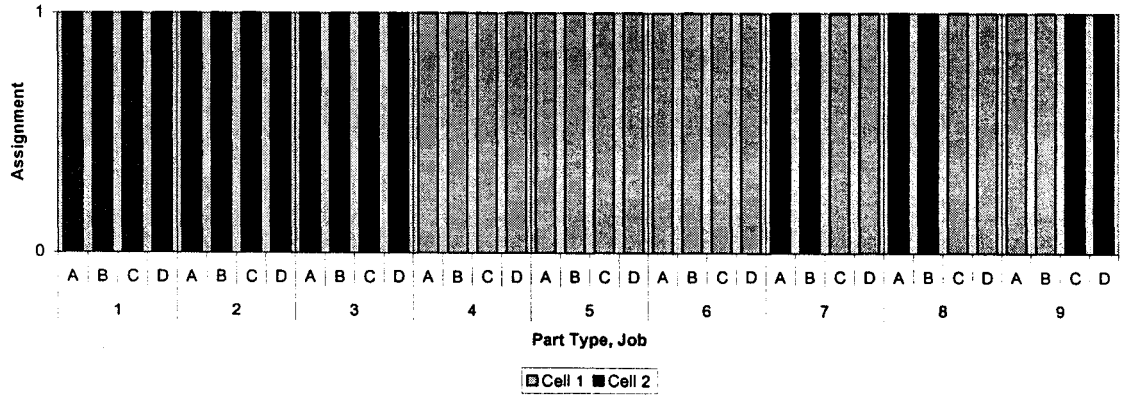

Figure 5.21: Part families

Table 5.31: Final cost values

Cost Function	Cost Value *($)*
Machine Capital Cost	266,000.00
Machine Idle Time Cost	18,298.33
Intercellular Movements Cost	24,650.00
Operator Cost	180,000.00
Operator Idle Time Cost	579.33
Lifting Risk Cost	200,273.83
Total Cost	**689,801.49**

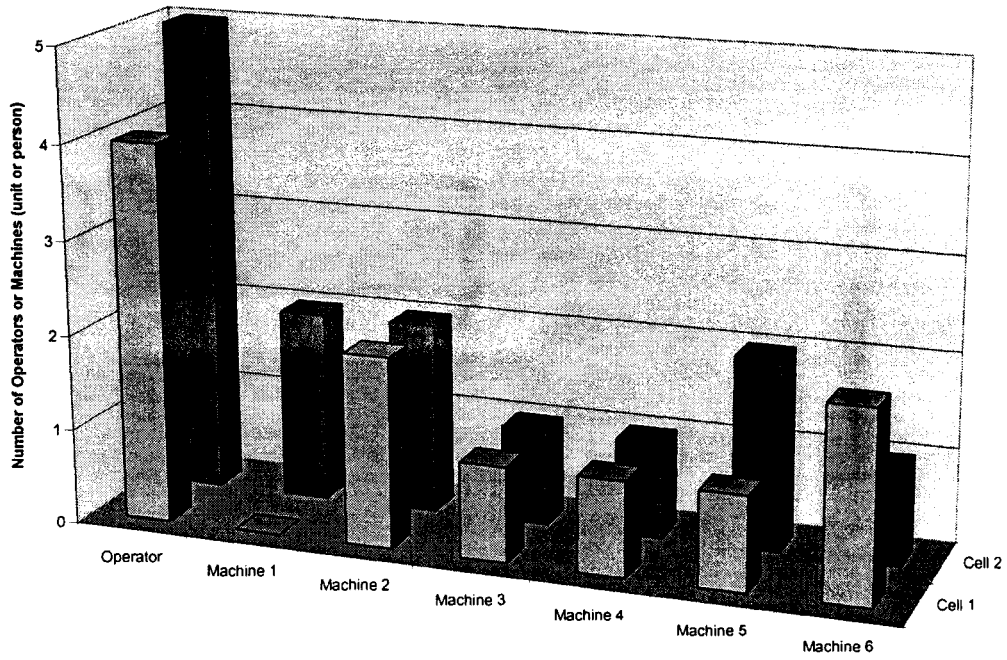

Figure 5.22: Operator and machine assignment

5.2.1.4. Comparison of Results of Example 3

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 3 has:

- 400 total variables, including 230 integer variables;
- 297 total constraints, including 151 nonlinear constraints; and
- 1226 total nonzeros, including 510 nonlinear nonzeros.

The SeqBBH method is not the best in terms of both running time and objective function value. The SimBBH method cannot solve this example. This situation happens because the nonlinearity in the model makes the solver stuck in infeasibility state. The

SimGA method solved the problem in the least running time and also gave the best system cost.

So, if SimGA method compared with SeqBBH method, the reduction in computational time is 56.60% (341 seconds to 148 seconds), and the reduction in the system cost is 11.46% ($779,102.46 to $689,801.49). The detailed comparison of the three methods is given in Table 5.32.

Table 5.32: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: Local Optimal	No Feasible Solution	Best Solution found after 687th generation
Solver Time	00:05:41	00:06:48	00:02:28
Cost Elements:			
Machine Capital Cost	*266,000.00*	-	*266,000.00*
Machine Idle Time Cost	*18,298.33*	-	*18,298.33*
Intercellular Movements Cost	*13,300.00*	-	*24,650.00*
Operator Cost	*220,000.00*	-	*180,000.00*
Operator Idle Time Cost	*16,579.33*	-	*579.33*
Lifting Risk Cost	*244,924.80*	-	*200,273.83*
Total System Cost	779,102.46	-	689,801.49

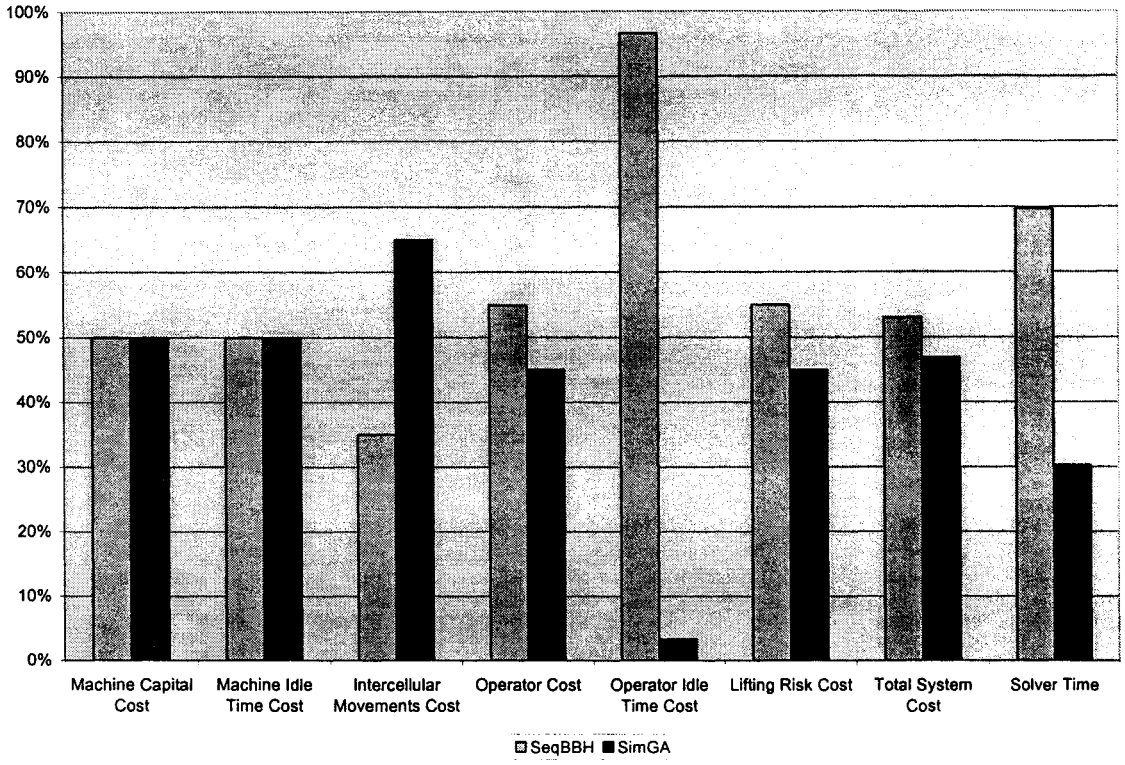

Figure 5.23: Costs and time comparisons

5.2.2. Example 4: Model 2 for Three-Cell Formation

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is \$10 per hour, operator idle time cost is \$4 per hour, and the approximate lifting risk compensation is \$15,000 per operator per lifting index. The cell capacity for machine cell 1 is 7 machines, for cell 2 is 14 machines, and for cell 3 is 12 machines. The maximum number of operators allowed for cell 1 is 7, and for cell 2 and cell 3 is 10.

The annual demand, load weight, vertical manual lifting distance, and intercellular movement cost for each part are given in Table 5.33. The machine

requirements and the machine information are given in Table 5.34 and Table 5.35, respectively.

Table 5.33: Annual demand, load weight, and vertical lifting distance

Part Type	Demand *(unit)*	Load Weight *(kg)*	Vertical Lifting Distance *(cm)*	Intercellular Movement Cost *($)*
1	26000	15	57	0.3
2	28000	9	41	0.5
3	17000	11	49	0.4
4	27000	7	54	0.35
5	29000	12	40	0.25
6	19000	17	45	0.4
7	25000	10	52	0.45
8	24000	8	36	0.3
9	16000	9	51	0.35
10	26000	12	42	0.4
11	16000	11	40	0.3
12	18000	17	66	0.5
13	24000	15	57	0.4
14	22000	6	52	0.35
15	28000	11	43	0.25
16	20000	13	41	0.35
17	30000	14	45	0.25
18	24000	21	30	0.4
19	18000	10	37	0.45
20	19000	9	48	0.5

Table 5.34: Machine requirements

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 1	Job A	0	0	114	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	108	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 2	Job A	0	0	0	0	0	168	0	0	0	0	0	0
	Job B	0	138	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 3	Job A	0	0	0	0	132	0	0	0	0	0	0	0
	Job B	0	132	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	114	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 4	Job A	0	0	0	0	0	0	0	0	0	126	0	0
	Job B	0	162	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	186	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 5	Job A	0	0	0	0	96	0	0	0	0	0	0	0
	Job B	0	0	102	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	108	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 6	Job A	174	0	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	108	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	150	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 7	Job A	0	0	0	0	0	0	84	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	0	108	0
	Job C	0	0	0	102	0	0	0	0	0	0	0	0
	Job D	174	0	0	0	0	0	0	0	0	0	0	0
Part 8	Job A	0	0	108	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	0	0	168
	Job C	0	0	0	0	0	0	174	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 9	Job A	0	0	0	0	0	0	168	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	174	0	0
	Job C	0	102	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 10	Job A	90	0	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	108	0	0
	Job C	0	0	0	108	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 11	Job A	0	0	0	0	102	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	126	0	0	0
	Job C	0	0	90	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 12	Job A	0	0	0	96	0	0	0	0	0	0	0	0
	Job B	120	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	144	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 11	Job A	0	0	126	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	138	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 12	Job A	0	0	0	96	0	0	0	0	0	0	0	0
	Job B	120	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	144	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 11	Job A	0	0	126	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	138	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 12	Job A	0	0	0	114	0	0	0	0	0	0	0	0
	Job B	0	150	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	120	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 15	Job A	0	0	150	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	0	0	192
	Job C	0	0	0	0	0	0	186	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 16	Job A	138	0	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	168	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	174	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 17	Job A	0	0	0	126	0	0	0	0	0	0	0	0
	Job B	0	0	174	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	162	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 18	Job A	78	0	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	102	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 19	Job A	102	0	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	120	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	180	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 20	Job A	0	0	0	0	0	192	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	96	0	0
	Job C	0	126	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0

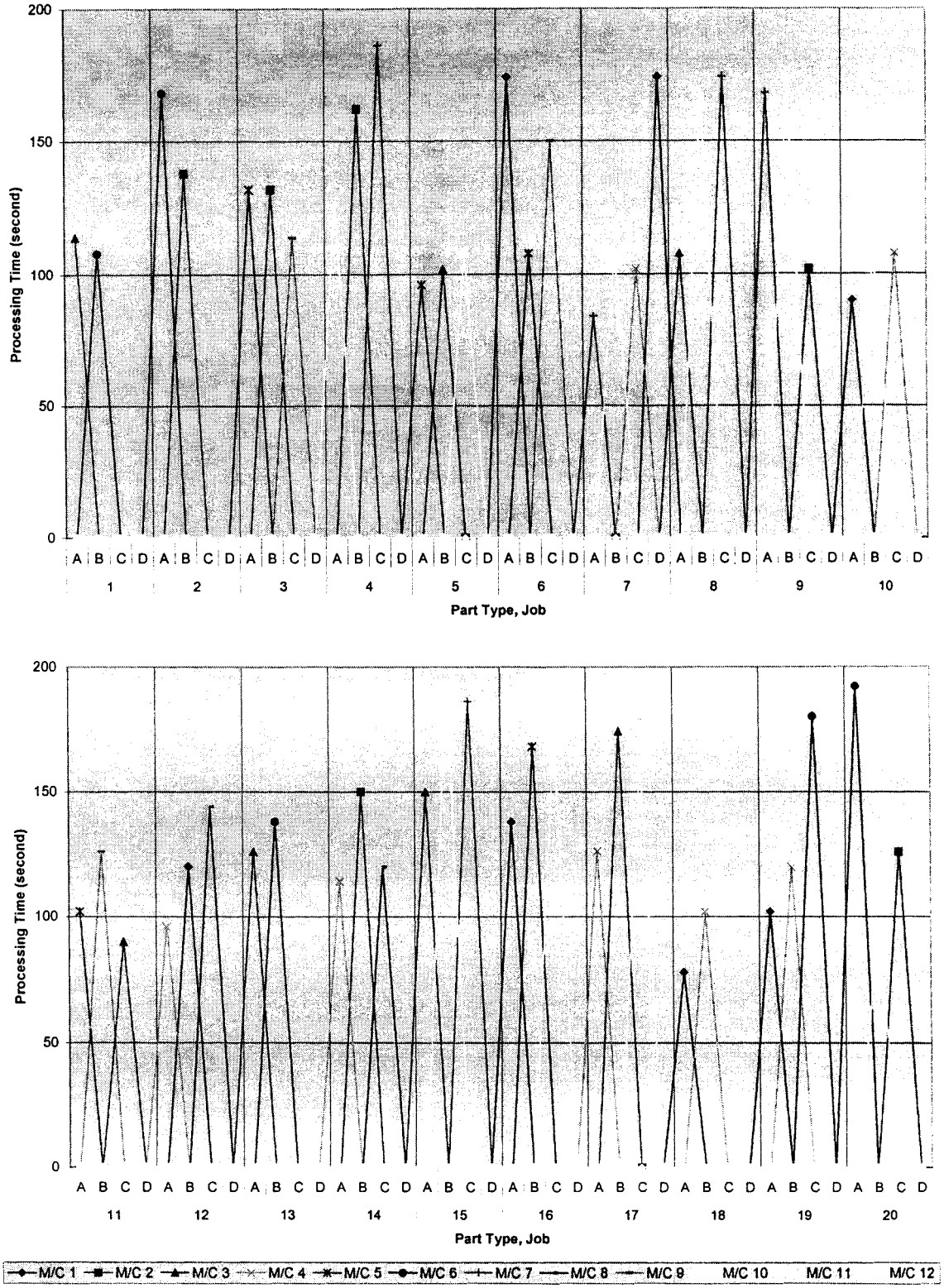

Figure 5.24: Processing time of each job on each part

Table 5.35: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	25000	0.75	4
2	26000	0.6	6
3	23000	0.75	5
4	15000	0.7	6
5	15000	0.65	3
6	24000	0.55	4
7	20000	0.75	4
8	18000	0.7	6
9	17000	0.5	5
10	16000	0.55	6
11	22000	0.5	5
12	21000	0.55	3

5.2.2.1. Result of SeqBBH Method

Trials have been made to solve Example 4 sequentially: Step 1, part families and machine cells formations are solved in 10 hours, 45 minutes, and 19 seconds with global optimal state; and Step 2, operator assignment which is after 34 minutes and 7 seconds, the LINGO solver could not find a feasible solution. This happens because after the Step 1, the part families and machine cells formations obtained make the model loses its flexibility to find a feasible solution on Step 2.

5.2.2.2. Result of SimBBH Method

Trials have been made to solve Example 4 simultaneously for the part families, machine cells formations, and operator assignment. However, after 1 hour, 9 minutes, and 59 seconds, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.2.2.3. Result of SimGA Method (GA2)

Example 4 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA2. The example is run for 4500 generations, with the population of 40 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution obtained after 2698th generation and the total time consumed is 32 minutes 15 seconds (see Figure 5.25).

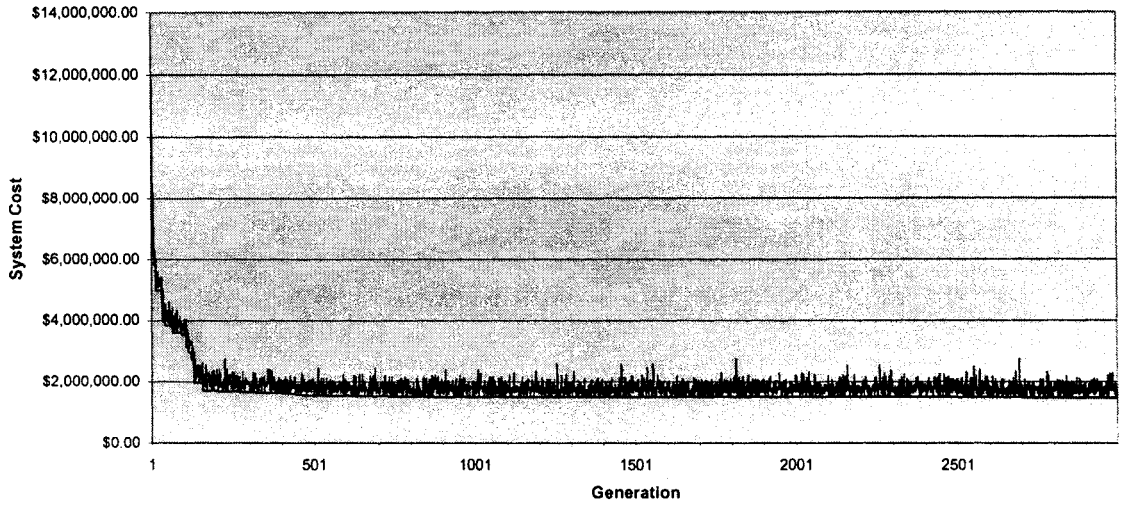

Figure 5.25: Average population cost and best cost for each generation

The first cell contains 1 unit of each machine type 4, 5, 7, 11, and 12. The second cell contains: 2 units of each machine type 1, 6, 7, and 10; 3 units of machine type 3; and 1 unit of each machine type 4, 5, and 9. The third cell contains: 1 unit of each machine type 1, 3, 4, 6, 8, 10, 11, and 12; and 3 units of machine type 2.

For part families, the first part family consists of part type 3, 4, 5, 7, 10, 11, 15, and 17. The second part family consists of part type 1, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15,

16, 17, 19, and 20. The third part family consists of part type 2, 3, 4, 8, 9, 10, 12, 14, 17, 18, 19, and 20. Intercellular movements occur on part type 3, 4, 5, 7, 8, 9, 10, 11, 12, 15, 17, 19, and 20.

For the first cell, the number of operators assigned is 3, the average frequency of lifting is 0.697 lifts per minute, and the composite lifting index is 1.115. For the second cell, the number of operators assigned is 8, the average lifting frequency is 0.630 lifts per minute, and the composite lifting index is 1.382. For the third cell, the number of operators assigned is 5, the average lifting frequency is 0.725 lifts per minute, and the composite lifting index is 1.495.

The results of part families and machine cells formations are summarized in Table 5.36. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.37. Also, various cost results of the solution are given in Table 5.38.

Table 5.36: Part families and machine cells formations

	Parts Assigned	Machines Assigned *(unit)*												
		1	2	3	4	5	6	7	8	9	10	11	12	Total
Cell 1	3A, 4C, 4D, 5B, 5C, 7A, 7B, 7C, 10D, 11A, 15B, 17A, 17D	0	0	0	1	1	0	1	0	0	0	1	1	5
Cell 2	1, 3C, 3D, 5B, 6, 7D, 8C, 8D, 9A, 9B, 10B, 10C, 11B, 11C, 11D, 12A, 12B, 13, 15A, 15C, 15D, 16, 17B, 19B, 19C, 19D, 20A	2	0	3	1	1	2	2	0	1	2	0	0	14
Cell 3	2, 3B, 4A, 4B, 8A, 8B, 9C, 9D, 10A, 12C, 12D, 14, 17C, 18, 19A, 20B, 20C, 20D	1	3	1	1	0	1	0	1	0	1	1	1	11

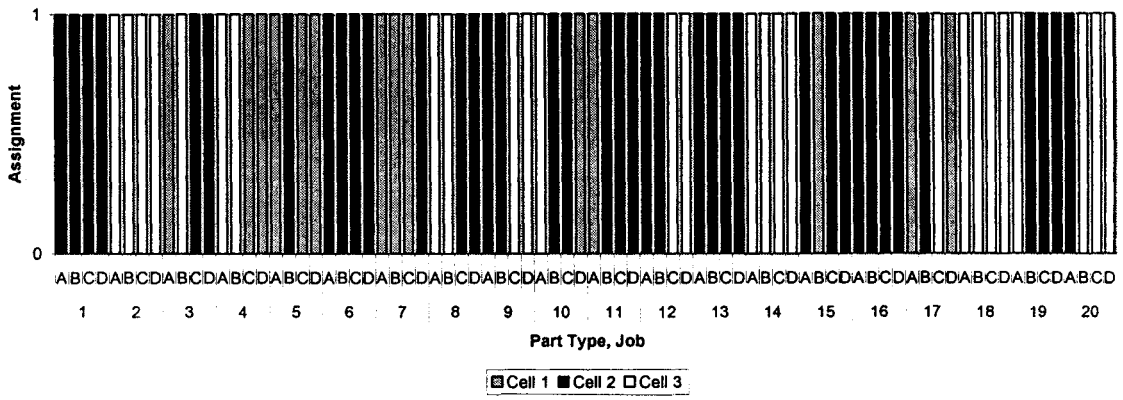
1
0
Assignment
A B C D A B C D A B C D

Table 5.38: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	621,000.00
Machine Idle Time Cost	58,791.67
Intercellular Movements Cost	150,300.00
Operator Cost	320,000.00
Operator Idle Time Cost	4,143.00
Lifting Risk Cost	328,147.42
Total Cost	**1,482,382.09**

5.2.2.4. Comparison of Results of Example 4

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 4 has:

- 1293 total variables, including 759 integer variables;
- 885 total constraints, including 490 nonlinear constraints; and
- 3918 total nonzeros, including 1653 nonlinear nonzeros.

The detailed comparison of the three methods is given in Table 5.39.

Table 5.39: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: No Feasible Solution	No Feasible Solution	Best Solution found 2698th generation
Solver Time	11:19:26	01:09:59	00:32:15
Cost Elements:			
Machine Capital Cost	-	-	*621,000.00*
Machine Idle Time Cost	-	-	*58,791.67*
Intercellular Movements Cost	-	-	*150,300.00*
Operator Cost	-	-	*320,000.00*
Operator Idle Time Cost	-	-	*4,143.00*
Lifting Risk Cost	-	-	*328,147.42*
Total System Cost	-	-	1,482,382.09

5.3. Model 3: Multi Period

In this section, Model 3 is tested using the three methods and each method with two numerical examples of different sizes. Some input data for the examples are taken from Ozdemir (1995). The first example considers 10 different part types, 7 different machine types, and 2 cells are to be formed in 3 periods of planning. The second example considers 20 different part types, 12 different machine types, and 3 cells are to be formed in 4 periods of planning.

5.3.1. Example 5: Model 3 for Two-Cell Formation and Three-Period Planning

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is $10 per hour, operator idle time cost is $4 per hour, and the approximate lifting risk compensation is $20,000 per operator per lifting index.

The annual demands for each period, load weight, and vertical manual lifting distance for each part are given in Table 5.40. The machine requirements and the machine information are given in Table 5.41 and Table 5.42, respectively. The cell capacities for each period are given in Table 5.43. The machine relocation cost and manpower level change cost are given in Table 5.44.

Table 5.40: Annual demands for each period, load weight, and vertical lifting distance

Part Type	Demand *(unit)*			Load Weight *(kg)*	Vertical Lifting Distance *(cm)*
	Period 1	Period 2	Period 3		
1	21000	22000	21000	12	37
2	19000	16000	13000	9	41
3	20000	18000	14000	11	49
4	20000	21000	18000	7	54
5	22000	23000	19000	14	40
6	23000	9000	6000	16	45
7	22000	19000	15000	10	52
8	19000	20000	21000	8	36
9	18000	20000	15000	9	51
10	20000	9000	7000	12	42

Table 5.41: Machine requirements

Part Type	Machining Time *(second)*						
	M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7
1	120	0	114	0	84	0	0
2	0	0	0	102	0	132	114
3	0	0	0	126	0	0	120
4	0	108	0	138	0	0	0
5	0	90	0	132	0	0	0
6	0	150	0	0	114	0	108
7	0	0	0	120	0	102	0
8	126	0	144	0	0	0	0
9	102	0	0	144	0	0	0
10	0	162	0	0	108	90	0

Table 5.42: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	18000	0.3	4
2	20000	0.4	6
3	22000	0.35	5
4	24000	0.55	6
5	23000	0.5	3
6	19000	0.65	4
7	20000	0.6	4

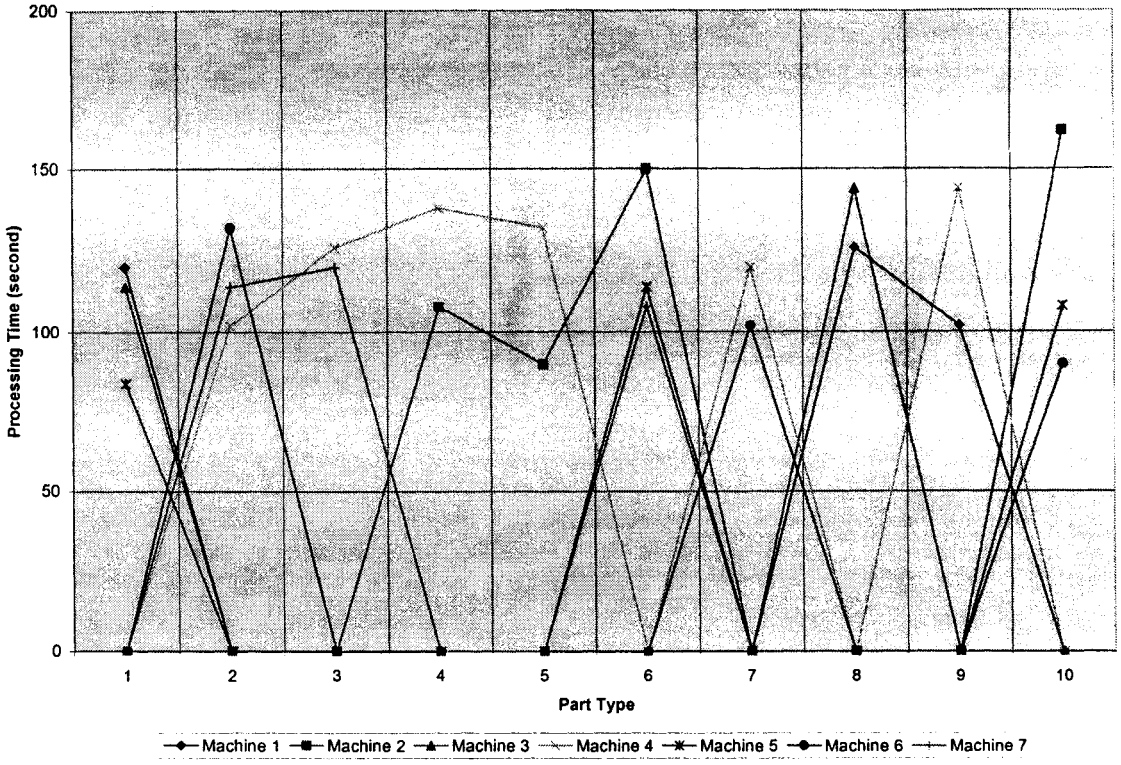

Figure 5.28: Processing time of each part

Table 5.43: Maximum number of machines and operators allowed (cell capacity)

	Max. number of machines			Max. number of operators		
	Period 1	Period 2	Period 3	Period 1	Period 2	Period 3
Cell 1	6	5	5	4	3	3
Cell 2	6	7	6	5	4	3

Table 5.44: Machine relocation cost and manpower level change cost

Change	Period 1	Period 2	Period 3
Machine Increase	5000	6000	3000
Machine Decrease	4000	5000	5000
Operator Increase	600	400	700
Operator Decrease	600	500	300

5.3.1.1. Result of SeqBBH Method

Example 5 is solved sequentially: Step 1, part families and machine cells formations are solved in 8 minutes and 9 seconds with global optimal state; and Step 2, operator assignment is solved in 2 minutes and 12 second with local optimal state.

The results of part families and machine cells formations are summarized in Table 5.45. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.46. Also, various cost results of the solution are given in Table 5.47.

Table 5.45: Part families and machine cells formations

		Parts Assigned	Machines Assigned *(unit)*							
			1	2	3	4	5	6	7	Total
Period 1	Cell 1	1, 4, 5, 8, 9	1	1	1	2	1	0	0	6
	Cell 2	2, 3, 6, 7, 10	0	1	0	1	1	1	1	5
Period 2	Cell 1	1, 8, 9	1	0	1	1	1	0	0	4
	Cell 2	2, 3, 4, 5, 6, 7, 10	0	1	0	2	1	1	1	6
Period 3	Cell 1	1, 8, 9	1	0	1	1	1	0	0	4
	Cell 2	2, 3, 4, 5, 6, 7, 10	0	1	0	2	1	1	1	6

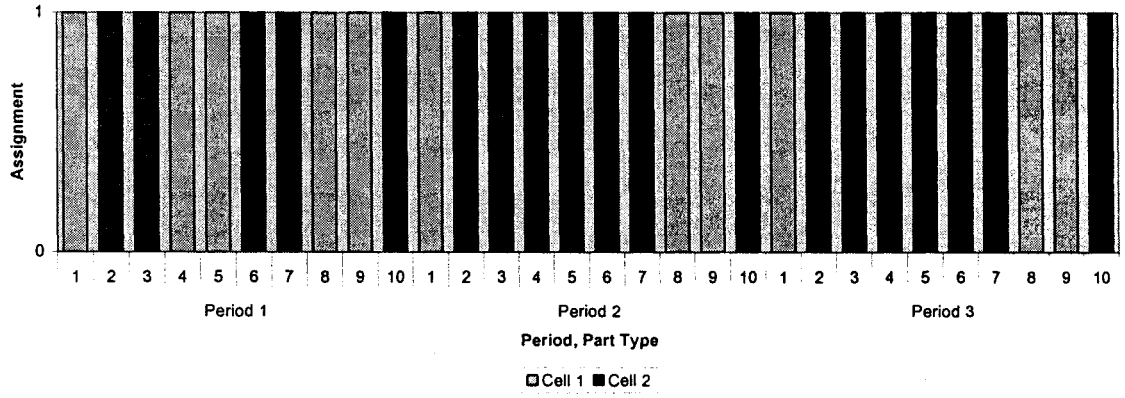

Figure 5.29: Part families

Table 5.46: Operator assignment, average frequency of lifting, and the composite lifting index

		Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Period 1	Cell 1	2	0.921	1.089
	Cell 2	3	0.750	1.264
Period 2	Cell 1	1	1.217	0.974
	Cell 2	3	0.733	1.249
Period 3	Cell 1	1	1.125	0.971
	Cell 2	2	0.875	1.250

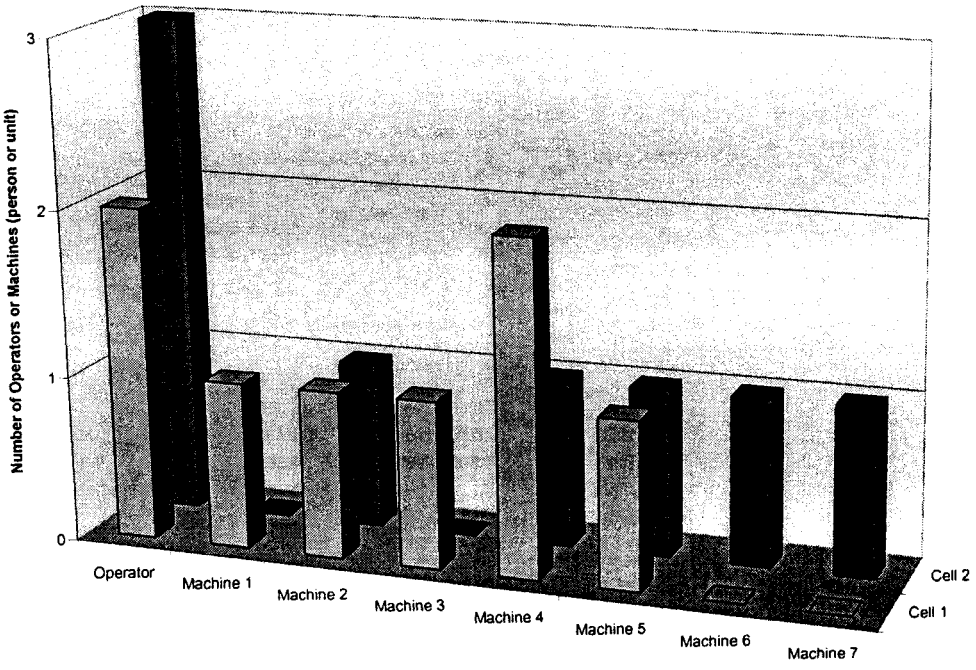

Figure 5.30: Operator and machine assignment for period 1

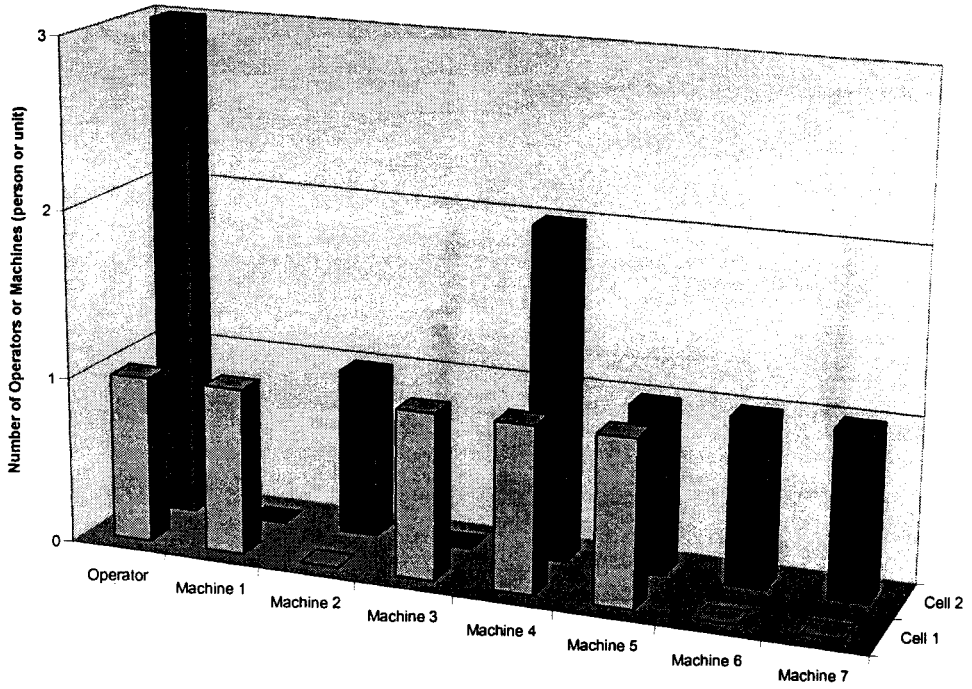

Figure 5.31: Operator and machine assignment for period 2

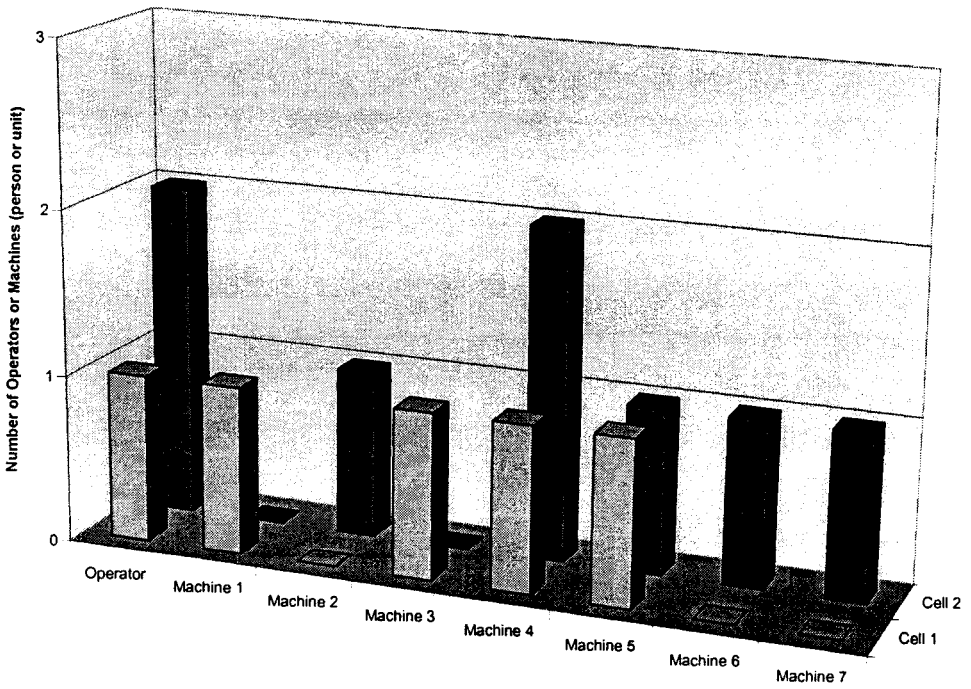

Figure 5.32: Operator and machine assignment for period 3

Table 5.47: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	671,000.00
Machine Idle Time Cost	92,946.67
Operator Cost	240,000.00
Operator Idle Time Cost	17,359.67
Lifting Risk Cost	283,270.40
Machine Relocation Cost	16,000.00
Manpower Level Change Cost	800.00
Total Cost	**1,321,376.74**

5.3.1.2. Result of SimBBH Method

Trials have been made to solve Example 5 simultaneously for the part families, machine cells formations, and operator assignment. However, after 22 hours, 20 minutes, and 34 seconds, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.3.1.3. Result of SimGA Method (GA3)

Example 5 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA3. The example is run for 3000 generations, with the population of 10 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 615^{th} generation and the total time consumed is 6 minutes and 3 seconds (see Figure 5.33).

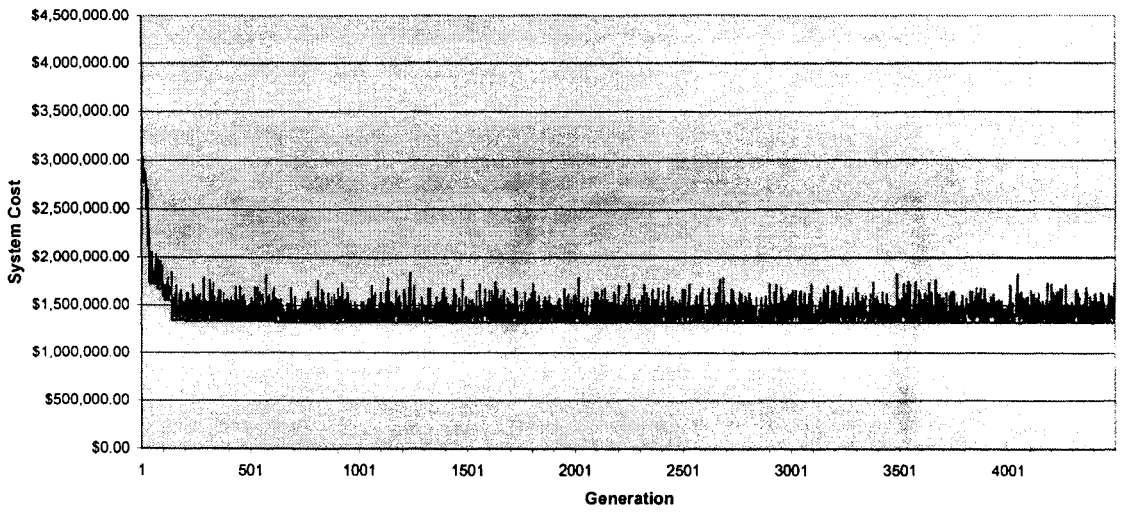

Figure 5.33: Average population cost and best cost for each generation

The results of part families and machine cells formations are summarized in Table 5.48. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.49. Also, various cost results of the solution are given in Table 5.50.

Table 5.48: Part families and machine cells formations

		Parts Assigned	Machines Assigned *(unit)*							
			1	2	3	4	5	6	7	Total
Period 1	Cell 1	1, 4, 5, 8, 9	1	1	1	2	1	0	0	6
	Cell 2	2, 3, 6, 7, 10	0	1	0	1	1	1	1	5
Period 2	Cell 1	1, 8, 9	1	0	1	1	1	0	0	4
	Cell 2	2, 3, 4, 5, 6, 7, 10	0	1	0	2	1	1	1	6
Period 3	Cell 1	1, 8, 9	1	0	1	1	1	0	0	4
	Cell 2	2, 3, 4, 5, 6, 7, 10	0	1	0	2	1	1	1	6

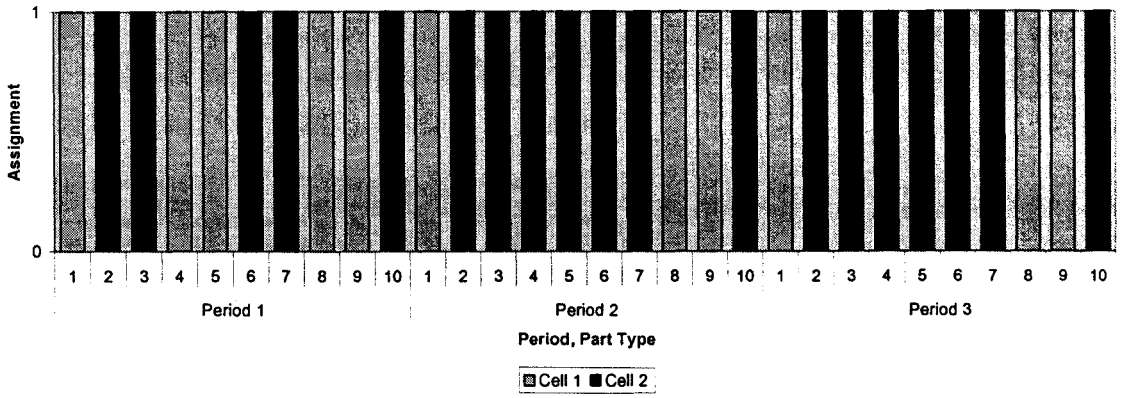

Figure 5.34: Part families

Table 5.49: Operator assignment, average frequency of lifting, and the composite lifting index

		Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Period 1	Cell 1	2	0.921	1.089
	Cell 2	3	0.750	1.264
Period 2	Cell 1	1	1.217	0.974
	Cell 2	3	0.733	1.249
Period 3	Cell 1	1	1.125	0.971
	Cell 2	2	0.875	1.250

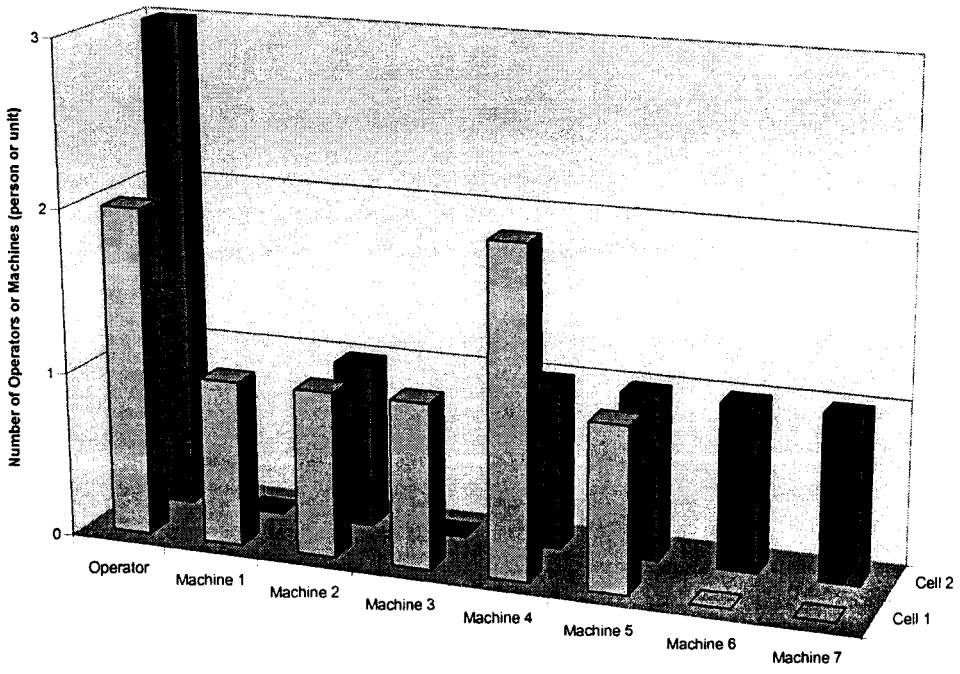

Figure 5.35: Operator and machine assignment for period 1

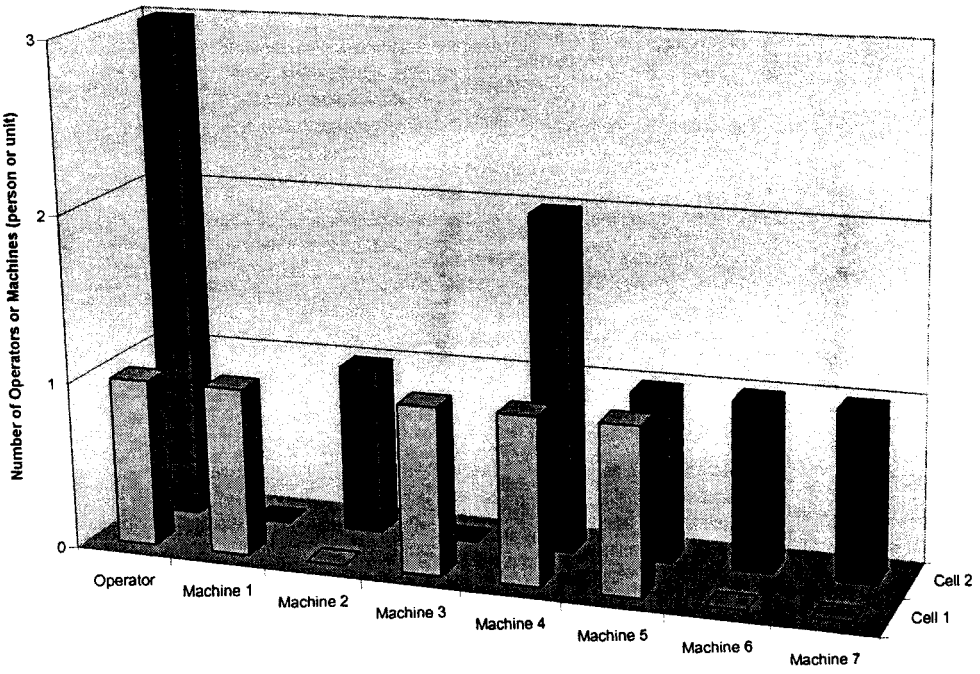

Figure 5.36: Operator and machine assignment for period 2

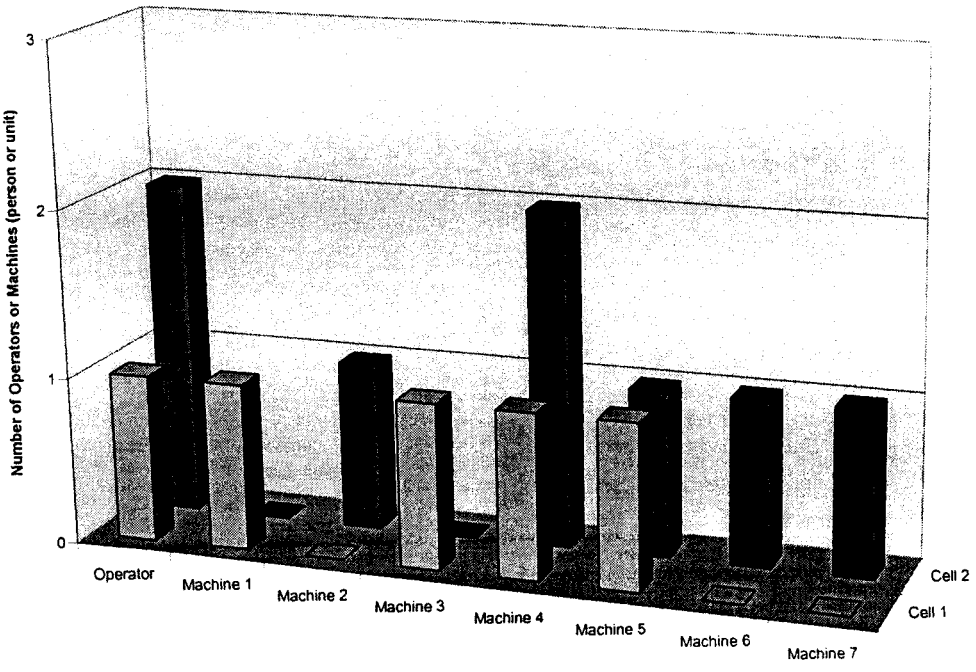

Figure 5.37: Operator and machine assignment for period 3

Table 5.50: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	671,000.00
Machine Idle Time Cost	92,946.67
Operator Cost	240,000.00
Operator Idle Time Cost	17,359.67
Lifting Risk Cost	283,270.40
Machine Relocation Cost	16,000.00
Manpower Level Change Cost	800.00
Total Cost	**1,321,376.74**

5.3.1.4. Comparison of Results of Example 5

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 5 has:

- 537 total variables, including 204 integer variables;
- 472 total constraints, including 139 nonlinear constraints; and
- 1588 total nonzeros, including 498 nonlinear nonzeros.

The SeqBBH method and the SimGA method gave the same system costs. But in terms of computational time, the SimGA method is faster than the SeqBBH method. The SimBBH method cannot solve this example. This situation happens because the nonlinearity in the model makes the solver stuck in infeasibility state.

So, if SimGA method compared with SeqBBH method, the reduction in computational time is 41.55% (621 seconds to 363 seconds), and the system cost values are the same. The detailed comparison of the three methods is given in Table 5.51.

Table 5.51: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: Local Optimal	No Feasible Solution	Best Solution found after 615th generation
Solver Time	00:10:21	22:20:34	00:06:03
Cost Elements:			
Machine Capital Cost	*671,000.00*	-	*671,000.00*
Machine Idle Time Cost	*92,946.67*	-	*92,946.67*
Operator Cost	*240,000.00*	-	*240,000.00*
Operator Idle Time Cost	*17,359.67*	-	*17,359.67*
Lifting Risk Cost	*283,270.40*	-	*283,270.40*
Machine Relocation Cost	*16,000.00*	-	*16,000.00*
Manpower Level Change Cost	*800.00*	-	*800.00*
Total System Cost	1,321,376.74	-	1,321,376.74

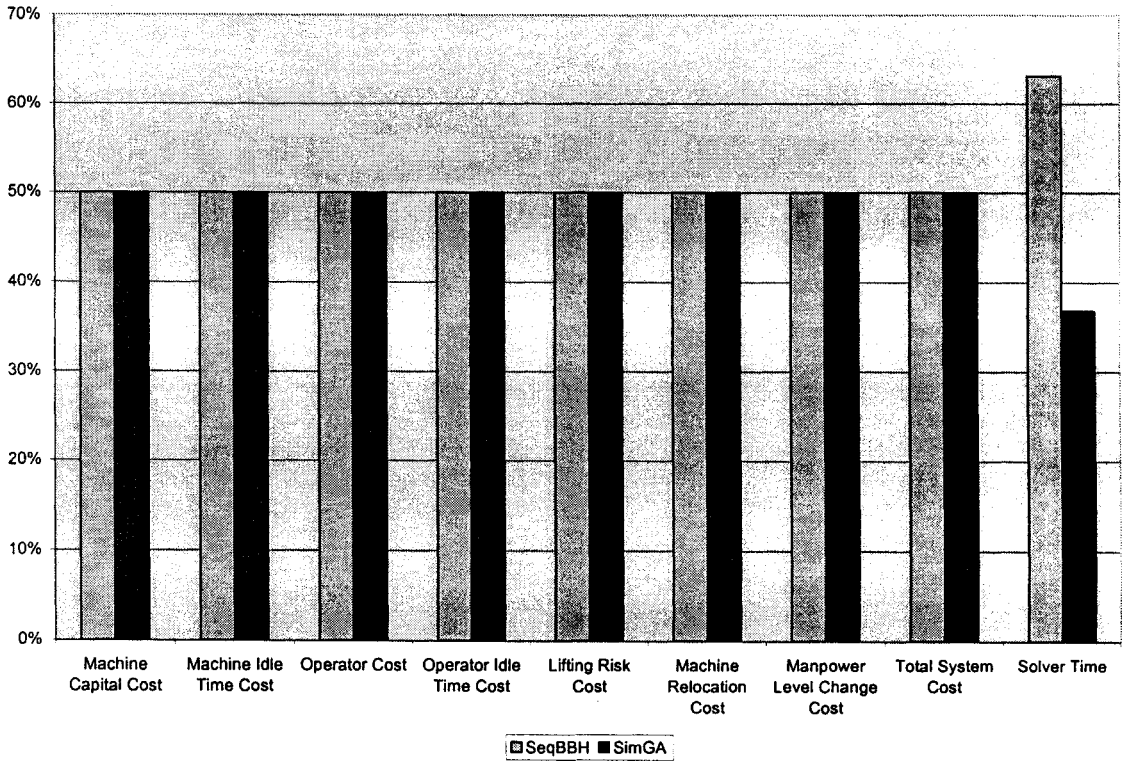

Figure 5.38: Costs and time comparisons

5.3.2. Example 6: Model 3 for Three-Cell Formation and Four-Period Planning

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is $10 per hour, operator idle time cost is $4 per hour, and the approximate lifting risk compensation is $20,000 per operator per lifting index.

The annual demands for each period, load weight, and vertical manual lifting distance for each part are given in Table 5.52. The machine requirements and the machine information are given in Table 5.53 and Table 5.54, respectively. The cell capacities for each period are given in Table 5.55. The machine relocation cost and manpower level change cost are given in Table 5.56.

Table 5.52: Annual demands for each period, load weight, and vertical lifting distance

Part Type	Demand *(unit)*				Load Weight *(kg)*	Vertical Lifting Distance *(cm)*
	Period 1	Period 2	Period 3	Period 4		
1	26000	22000	21000	19000	15	57
2	28000	16000	13000	9000	9	41
3	17000	18000	14000	8000	11	49
4	27000	21000	18000	17000	7	54
5	29000	23000	19000	15000	12	40
6	11000	9000	6000	9000	17	45
7	25000	19000	15000	19000	10	52
8	24000	20000	21000	20000	8	36
9	16000	20000	15000	20000	9	51
10	26000	9000	17000	19000	12	42
11	16000	13000	9000	15000	11	40
12	18000	15000	25000	16000	17	66
13	24000	10000	9000	12000	15	57
14	22000	16000	11000	8000	6	52
15	28000	14000	8000	16000	11	43
16	20000	22000	12000	12000	13	41
17	30000	24000	20000	28000	14	45
18	24000	20000	17000	12000	21	30
19	18000	19000	25000	17000	10	37
20	19000	22000	25000	20000	9	48

Table 5.53: Machine requirements

Part Type	Machining Time *(second)*											
	M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
1	0	0	114	0	0	108	0	0	126	0	0	0
2	0	72	0	0	0	168	0	144	0	0	180	0
3	0	0	0	126	132	0	0	0	0	0	0	0
4	0	126	0	0	0	0	0	0	138	126	0	0
5	0	72	0	0	96	0	0	0	0	114	0	0
6	174	0	0	0	156	0	0	0	0	0	0	0
7	0	90	0	0	0	0	84	0	0	0	0	192
8	0	0	108	0	0	0	0	132	0	150	0	0
9	0	0	0	0	0	0	168	0	108	0	102	126
10	90	0	0	0	0	0	138	162	0	0	72	0
11	0	0	0	72	102	0	78	0	108	0	0	0
12	0	126	0	96	0	0	0	0	0	0	0	78
13	0	0	126	0	0	0	0	0	0	144	90	0
14	0	0	0	114	0	0	0	0	0	0	0	126
15	0	102	150	0	0	150	0	0	0	0	0	0
16	138	0	78	0	0	0	0	72	0	0	192	0
17	0	0	0	126	0	0	132	0	120	0	0	0
18	78	84	0	96	0	0	0	132	0	0	0	0
19	102	0	0	0	0	0	0	0	0	192	0	0
20	0	0	90	0	120	192	0	0	0	0	0	108

Table 5.54: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	25000	0.75	4
2	26000	0.6	6
3	23000	0.75	5
4	15000	0.7	6
5	15000	0.65	3
6	24000	0.55	4
7	20000	0.75	4
8	18000	0.7	6
9	17000	0.5	5
10	16000	0.55	6
11	22000	0.5	5
12	21000	0.55	3

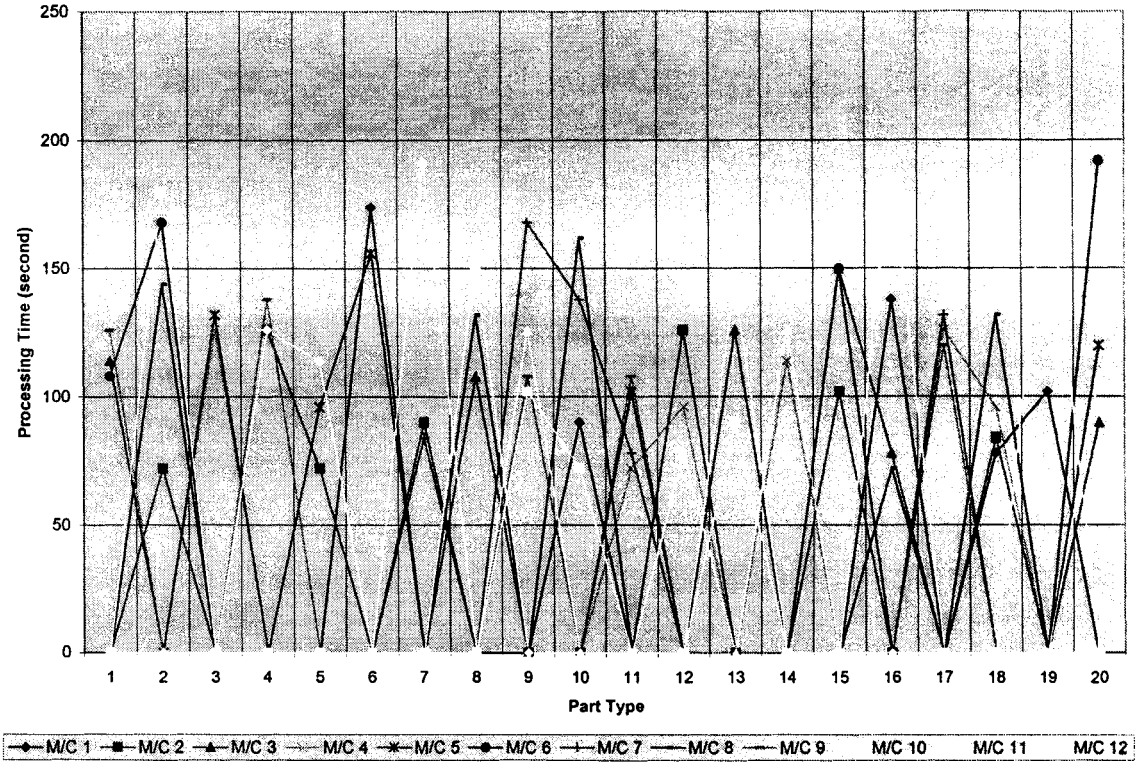

Figure 5.39: Processing time for each part

Table 5.55: Maximum number of machines and operators allowed (cell capacity)

	Max. number of machines				Max. number of operators			
	Period 1	Period 2	Period 3	Period 4	Period 1	Period 2	Period 3	Period 4
Cell 1	15	15	14	12	8	9	9	8
Cell 2	12	11	11	10	9	8	8	7
Cell 3	12	12	12	12	9	9	8	7

Table 5.56: Machine relocation cost and manpower level change cost

	Period 1	Period 2	Period 3	Period 4
Machine Increase	4000	6000	3000	4000
Machine Decrease	5000	4000	5000	4500
Operator Increase	600	400	700	650
Operator Decrease	600	500	300	400

5.3.2.1. Result of SeqBBH Method

Trials have been made to solve Example 6 sequentially: Step 1, part families and machine cells formations are solved in 19 hours, 4 minutes, and 22 seconds with global optimal state; and Step 2, operator assignment which is after 3 minutes and 58 seconds, the LINGO solver could not find a feasible solution. This happens because after the Step 1, the part families and machine cells formations obtained make the model loses its flexibility to find a feasible solution on Step 2.

5.3.2.2. Result of SimBBH Method

Trials have been made to solve Example 6 simultaneously for the part families, machine cells formations, and operator assignment. However, after 2 hours, 15 minutes, and 27 seconds, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.3.2.3. Result of SimGA Method (GA3)

Example 6 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA3. The example is run for 4500 generations, with the population of 20 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 2300^{th} generation and the total time consumed is 39 minutes and 38 seconds (see Figure 5.40).

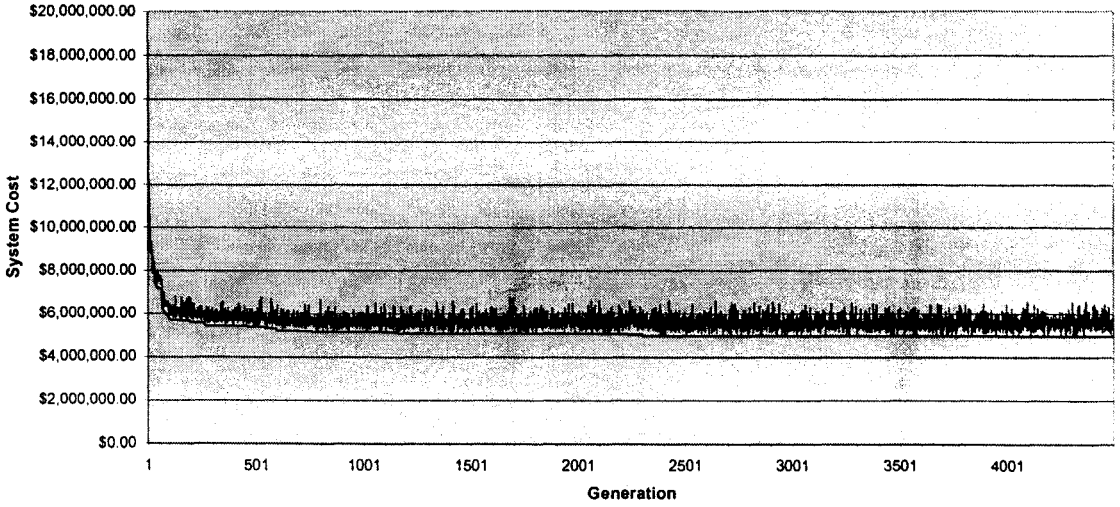

Figure 5.40: Average population cost and best cost for each generation

The results of part families and machine cells formations are summarized in Table 5.57 and Table 5.58, respectively. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.59. Also, various cost results of the solution are given in Table 5.60.

Table 5.57: Part families

		Parts Assigned
Period 1	Cell 1	1, 3, 5, 6, 10, 11, 12, 13, 16, 20
	Cell 2	4, 14, 18, 19
	Cell 3	2, 7, 8, 9, 15, 17
Period 2	Cell 1	1, 5, 6, 7, 8, 10, 11, 12, 13, 16, 17, 20
	Cell 2	-
	Cell 3	2, 3, 4, 9, 14, 15, 18, 19
Period 3	Cell 1	3, 4, 5, 8, 9, 14, 18, 20
	Cell 2	-
	Cell 3	1, 2, 6, 7, 10, 11, 12, 13, 15, 16, 17, 19
Period 4	Cell 1	1, 2, 3, 5, 6, 7, 8, 12, 13, 17
	Cell 2	-
	Cell 3	4, 9, 10, 11, 14, 15, 16, 18, 19, 20

Table 5.58: Machine cells formations

		Machines Assigned *(unit)*												
		1	2	3	4	5	6	7	8	9	10	11	12	Total
Period 1	Cell 1	1	1	2	1	2	1	1	1	1	1	2	1	15
	Cell 2	1	1	0	1	0	0	0	1	1	1	0	1	7
	Cell 3	0	1	1	1	0	2	2	1	1	1	1	1	12
Period 2	Cell 1	1	1	2	1	2	1	1	1	1	1	1	1	14
	Cell 2	0	0	0	0	0	0	0	0	0	0	0	0	0
	Cell 3	1	1	1	1	1	1	1	1	1	1	1	1	12
Period 3	Cell 1	1	1	1	1	1	1	1	1	1	2	1	1	13
	Cell 2	0	0	0	0	0	0	0	0	0	0	0	0	0
	Cell 3	1	1	1	1	1	1	1	1	1	1	1	1	12
Period 4	Cell 1	1	1	1	1	1	1	1	1	1	1	1	1	12
	Cell 2	0	0	0	0	0	0	0	0	0	0	0	0	0
	Cell 3	1	1	1	1	1	1	1	1	1	1	1	1	12

Table 5.59: Operator assignment, average frequency of lifting, and the composite lifting index

		Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Period 1	Cell 1	7	0.799	1.394
	Cell 2	3	0.714	1.493
	Cell 3	6	0.690	1.102
Period 2	Cell 1	7	0.811	1.385
	Cell 2	0	0	0
	Cell 3	5	0.725	1.495
Period 3	Cell 1	5	0.753	1.490
	Cell 2	0	0	0
	Cell 3	6	0.778	1.393
Period 4	Cell 1	5	0.762	1.389
	Cell 2	0	0	0
	Cell 3	6	0.751	1.498

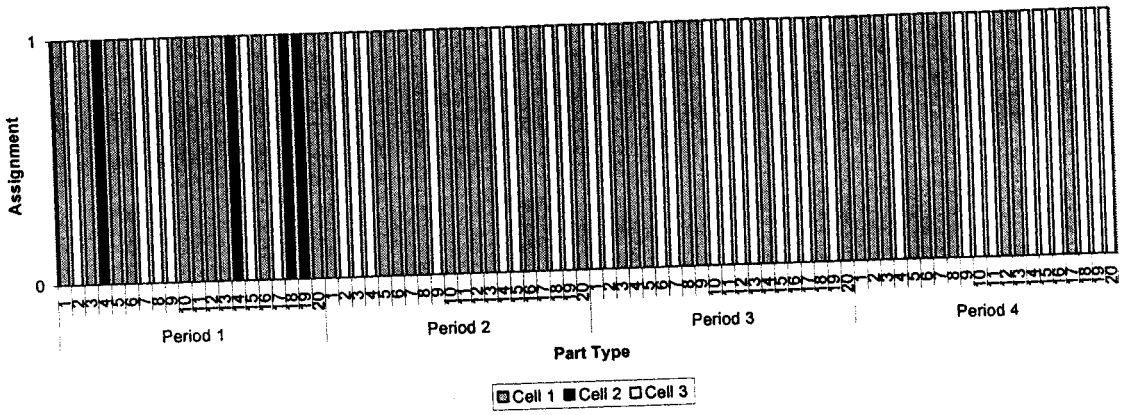

Figure 5.41: Part families

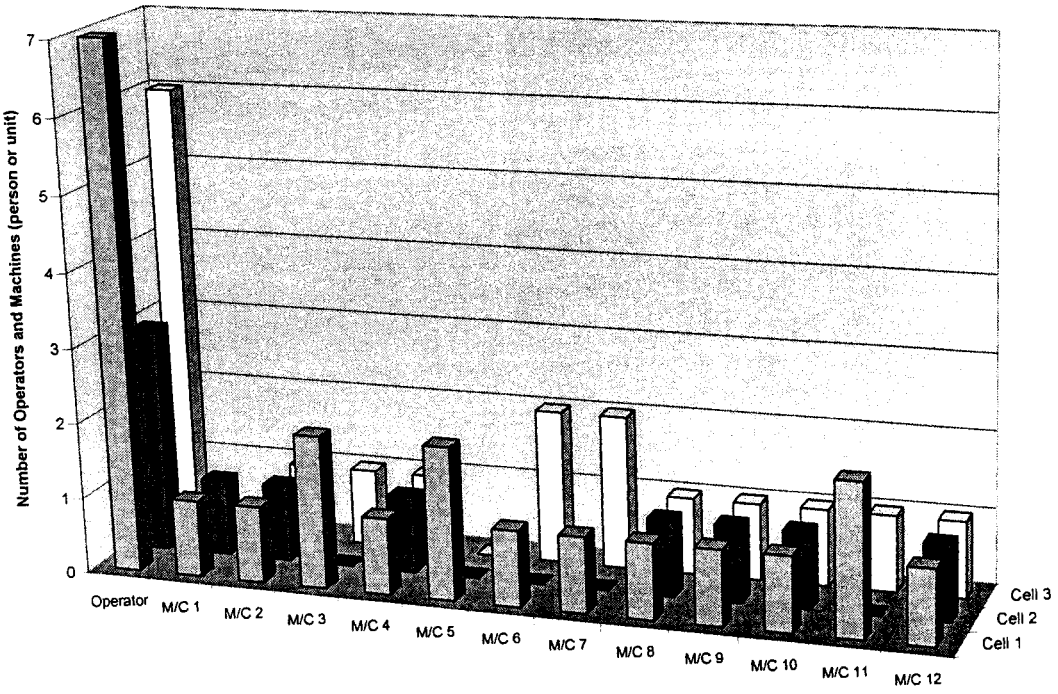

Figure 5.42: Operator and machine assignment for period 1

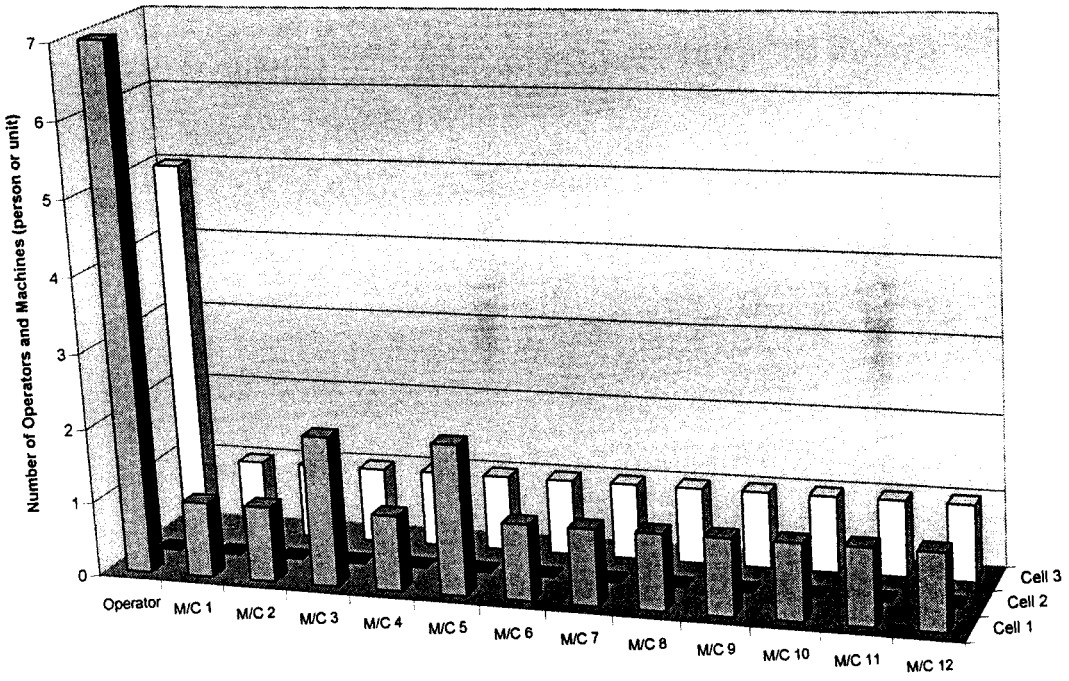

Figure 5.43: Operator and machine assignment for period 2

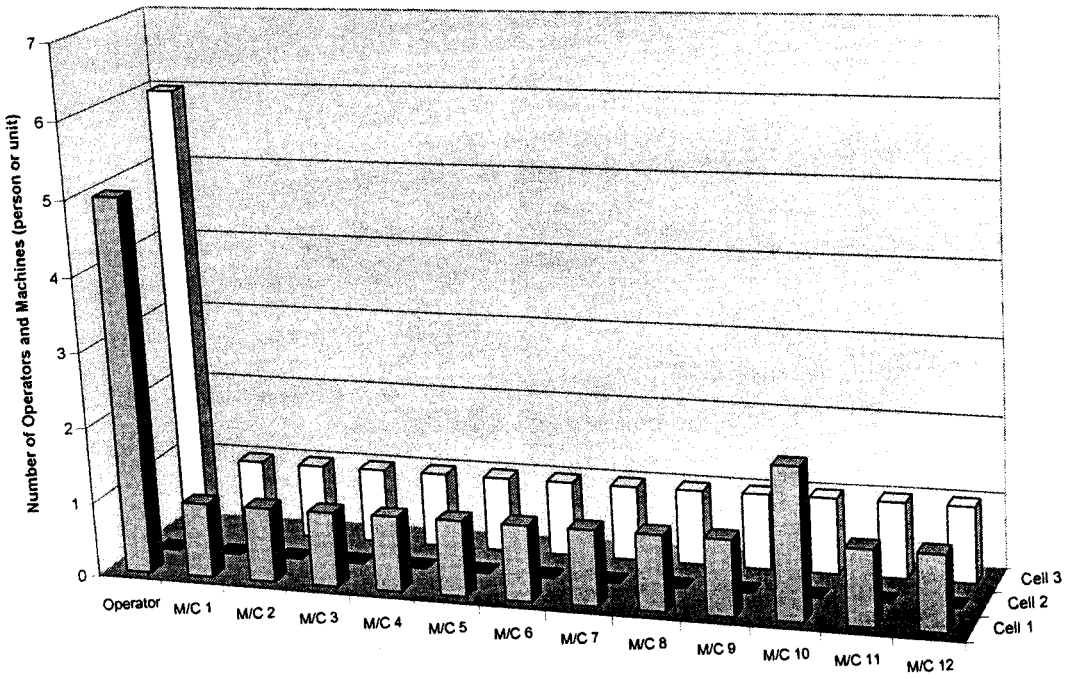

Figure 5.44: Operator and machine assignment for period 3

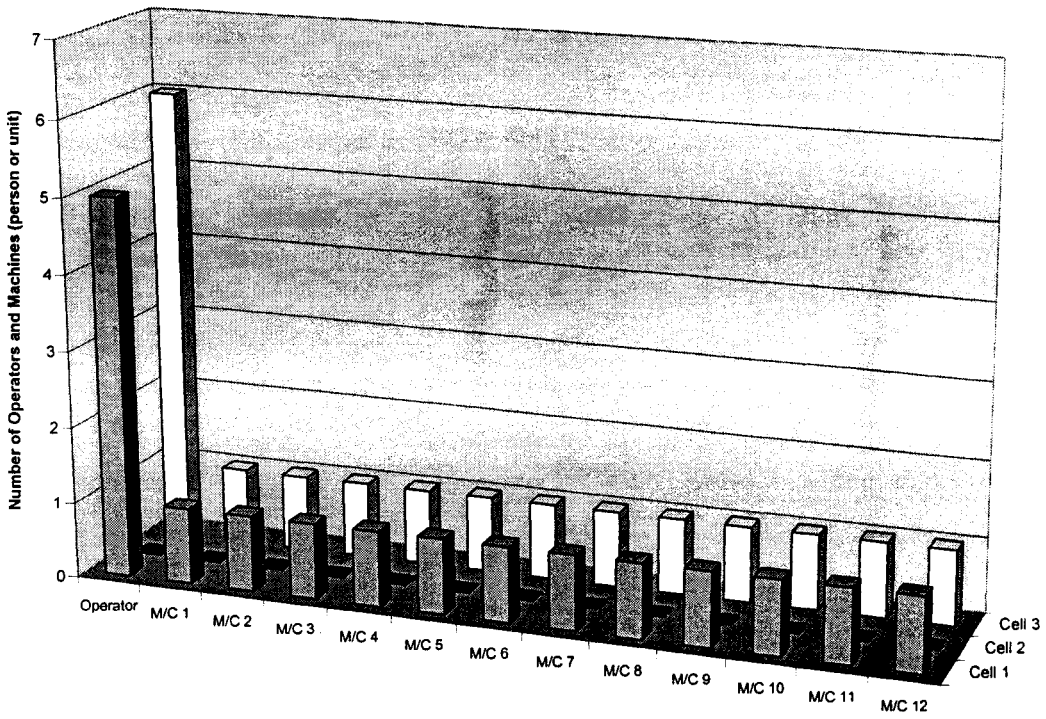

Figure 5.45: Operator and machine assignment for period 4

Table 5.60: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	2,192,000.00
Machine Idle Time Cost	305,618.33
Operator Cost	1,000,000.00
Operator Idle Time Cost	19,036.00
Lifting Risk Cost	1,395,231.91
Machine Relocation Cost	69,500.00
Manpower Level Change Cost	3,300.00
Total Cost	**4,984,686.24**

5.3.2.4. Comparison of Results of Example 6

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function value.

LINGO reported that Example 6 has:

- 1907 total variables, including 708 integer variables;
- 1549 total constraints, including 517 nonlinear constraints; and
- 5942 total nonzeros, including 1956 nonlinear nonzeros.

The detailed comparison of the three methods is given in Table 5.61.

Table 5.61: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: No Feasible Solution	No Feasible Solution	Best Solution found after 2300th generation
Solver Time	19:08:20	02:15:37	00:39:38
Cost Elements:			
Machine Capital Cost	-	-	*2,192,000.00*
Machine Idle Time Cost	-	-	*305,618.33*
Operator Cost	-	-	*1,000,000.00*
Operator Idle Time Cost	-	-	*19,036.00*
Lifting Risk Cost	-	-	*1,395,231.91*
Machine Relocation Cost	-	-	*69,500.00*
Manpower Level Change Cost	-	-	*3,300.00*
Total System Cost	-	-	4,984,686.24

5.4. Model 4: Multi Period with Job Sequence

In this section, Model 4 is tested using the three methods and each method with two numerical examples of different sizes. Some input data for the examples are taken from Ozdemir (1995). The first example considers 10 different part types, each part consists of 3 operations or less, 7 different machine types, and 2 cells are to be formed in 3 periods of planning. The second example considers 20 different part types, each part

consists of 4 operations or less, 12 different machine types, and 4 cells are to be formed in 4 periods of planning.

5.4.1. Example 7: Model 4 for Two-Cell Formation and Three-Period Planning

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is \$10 per hour, operator idle time cost is \$4 per hour, and the approximate lifting risk compensation is \$20,000 per operator per lifting index.

The annual demands for each period, load weight, vertical manual lifting distance and intercellular movement cost for each part are given in Table 5.62. The machine requirements and the machine information are given in Table 5.63 and Table 5.64, respectively. The cell capacities for each period are given in Table 5.65. The machine relocation cost and the manpower level change cost are given in Table 5.66.

Table 5.62: Annual demands for each period, load weight, vertical lifting distance, and intercellular movement cost

Part Type	Demand *(unit)*			Load Weight *(kg)*	Vertical Lifting Distance *(cm)*	Intercellular Movement Cost *($)*
	Period 1	Period 2	Period 3			
1	21000	22000	21000	12	37	0.3
2	19000	16000	13000	9	41	0.5
3	20000	18000	14000	11	49	0.4
4	20000	21000	18000	7	54	0.35
5	22000	23000	19000	14	40	0.25
6	23000	9000	6000	16	45	0.4
7	22000	19000	15000	10	52	0.45
8	19000	20000	21000	8	36	0.3
9	18000	20000	15000	9	51	0.35
10	20000	9000	7000	12	42	0.4

Table 5.63: Machine requirements

Part Type & Jobs		Machining Time *(second)*						
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7
Part 1	Job A	0	0	114	0	0	0	0
	Job B	120	0	0	0	0	0	0
	Job C	0	0	0	0	84	0	0
Part 2	Job A	0	0	0	102	0	0	0
	Job B	0	0	0	0	0	0	114
	Job C	0	0	0	0	0	132	0
Part 3	Job A	0	0	0	0	0	0	120
	Job B	0	0	0	126	0	0	0
	Job C	0	0	0	0	0	0	0
Part 4	Job A	0	0	0	138	0	0	0
	Job B	0	108	0	0	0	0	0
	Job C	0	0	0	0	0	0	0
Part 5	Job A	0	90	0	0	0	0	0
	Job B	0	0	0	132	0	0	0
	Job C	0	0	0	0	0	0	0
Part 6	Job A	0	0	0	0	0	0	108
	Job B	0	150	0	0	0	0	0
	Job C	0	0	0	0	114	0	0
Part 7	Job A	0	0	0	0	0	102	0
	Job B	0	0	0	120	0	0	0
	Job C	0	0	0	0	0	0	0
Part 8	Job A	126	0	0	0	0	0	0
	Job B	0	0	144	0	0	0	0
	Job C	0	0	0	0	0	0	0
Part 9	Job A	102	0	0	0	0	0	0
	Job B	0	0	0	144	0	0	0
	Job C	0	0	0	0	0	0	0
Part 10	Job A	0	0	0	0	0	90	0
	Job B	0	162	0	0	0	0	0
	Job C	0	0	0	0	108	0	0

Table 5.64: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	18000	0.3	4
2	20000	0.4	6
3	22000	0.35	5
4	24000	0.55	6
5	23000	0.5	3
6	19000	0.65	4
7	20000	0.6	4

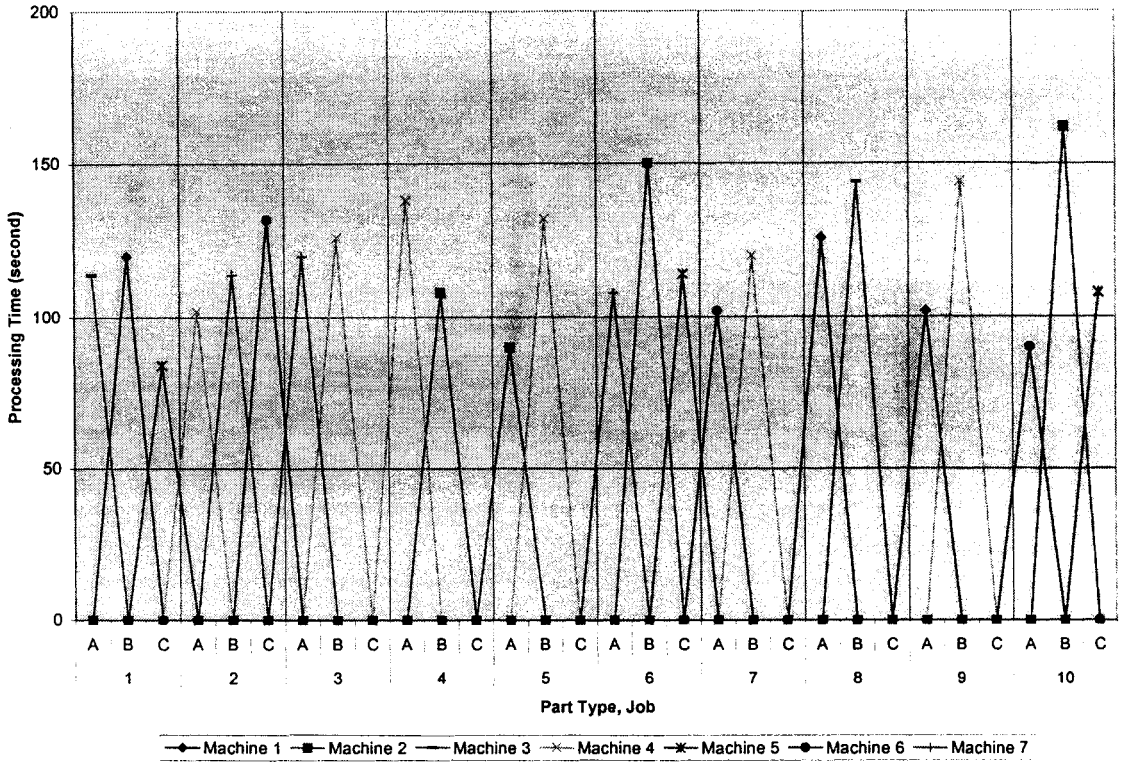

Figure 5.46: Processing time of each job on each part

Table 5.65: Maximum number of machines and operators allowed (cell capacity)

	Max. number of machines			Max. number of operators		
	Period 1	Period 2	Period 3	Period 1	Period 2	Period 3
Cell 1	10	8	8	6	5	5
Cell 2	8	8	9	5	4	5

Table 5.66: Machine relocation cost and manpower level change cost

	Period 1	Period 2	Period 3
Machine Increase	5000	6000	3000
Machine Decrease	4000	5000	5000
Operator Increase	600	400	700
Operator Decrease	600	500	300

5.4.1.1. Result of SeqBBH Method

Trials have been made to solve Example 7 sequentially: Step 1, part families and machine cells formations are solved in 14 hours, 28 minutes, and 11 seconds with global optimal state; and Step 2, operator assignment which is after 1 hour, 15 minutes, and 8 seconds, the LINGO solver could not find a feasible solution. This happens because after the Step 1, the part families and machine cells formations obtained make the model loses its flexibility to find a feasible solution on Step 2.

5.4.1.2. Result of SimBBH Method

Example 7 is solved simultaneously for the part families, machine cells formations, and operator assignment. This example is solved in 7 hours, 19 minutes, and 57 seconds with local optimal state.

The results of part families and machine cells formations are summarized in Table 5.67. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.68. Also, various cost results of the solution are given in Table 5.69.

Table 5.67: Part families and machine cells formations

		Parts Assigned	Machines Assigned *(unit)*							
			1	2	3	4	5	6	7	Total
Period 1	Cell 1	1, 4, 5, 6C, 10C	1	1	1	1	1	0	0	5
	Cell 2	2, 3, 6A, 6B, 7, 8, 9, 10A, 10B	1	1	1	2	0	1	1	7
Period 2	Cell 1	1, 5, 6B, 6C, 8, 10B, 10C	1	1	1	1	1	0	0	5
	Cell 2	2, 3, 4, 6A, 7, 9, 10A	1	1	0	2	0	1	1	6
Period 3	Cell 1	1, 4, 5, 6B, 6C, 8, 10B, 10C	1	1	1	1	1	0	0	5
	Cell 2	2, 6A, 7, 9, 10A	1	0	0	1	0	1	1	4

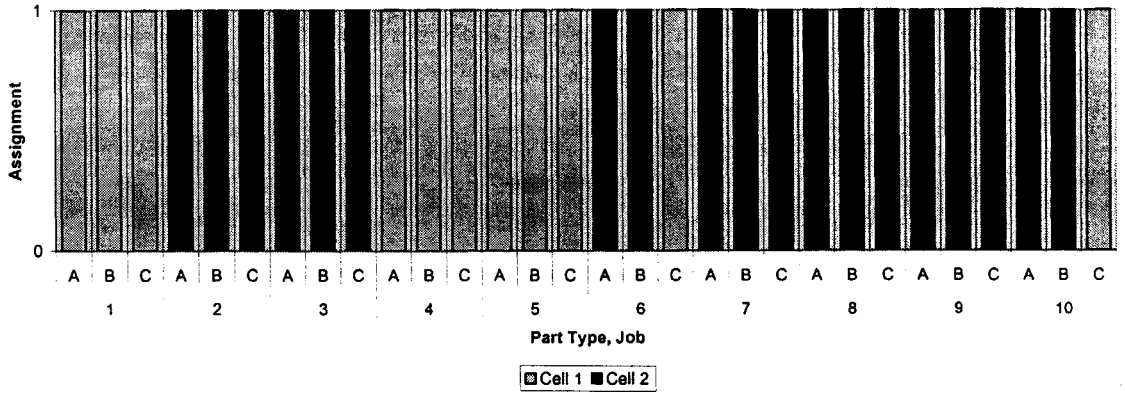

Figure 5.47: Part families in period 1

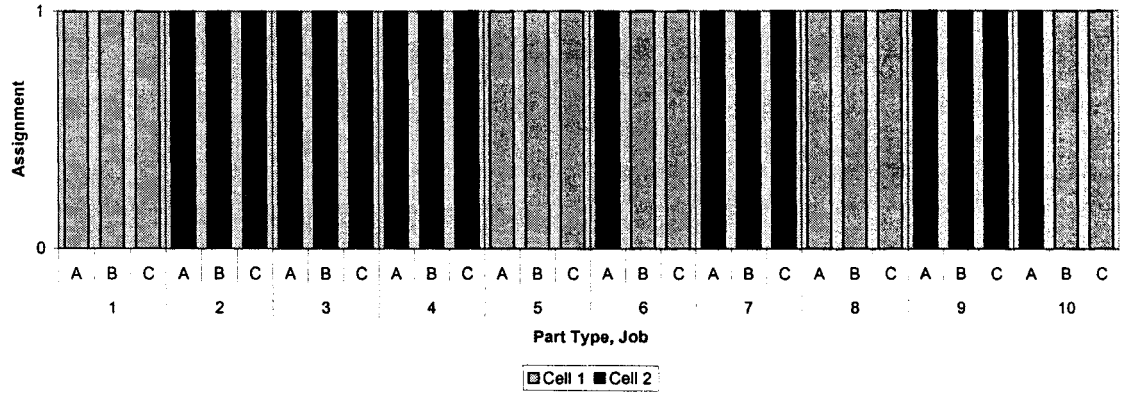

Figure 5.48: Part families in period 2

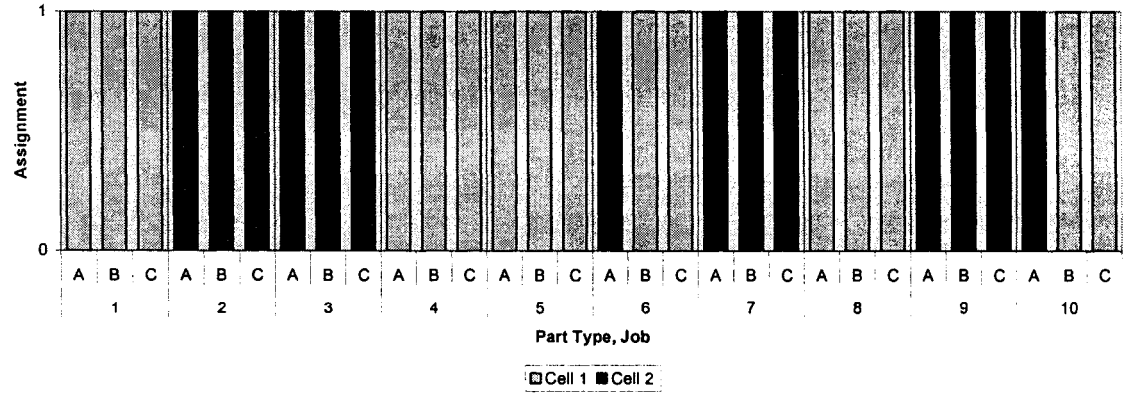

Figure 5.49: Part families in period 3

Table 5.68: Operator assignment, average frequency of lifting, and the composite lifting index

		Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Period 1	Cell 1	2	0.792	1.262
	Cell 2	3	0.836	1.242
Period 2	Cell 1	2	0.783	1.267
	Cell 2	2	0.925	1.227
Period 3	Cell 1	2	0.854	1.252
	Cell 2	2	0.583	1.233

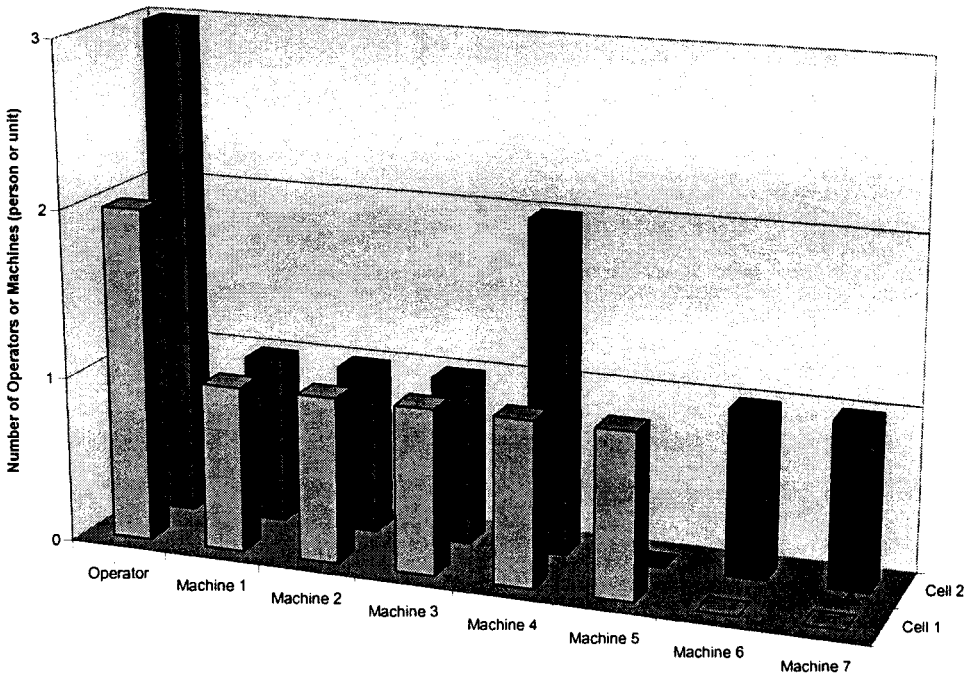

Figure 5.50: Operator and machine assignment in period 1

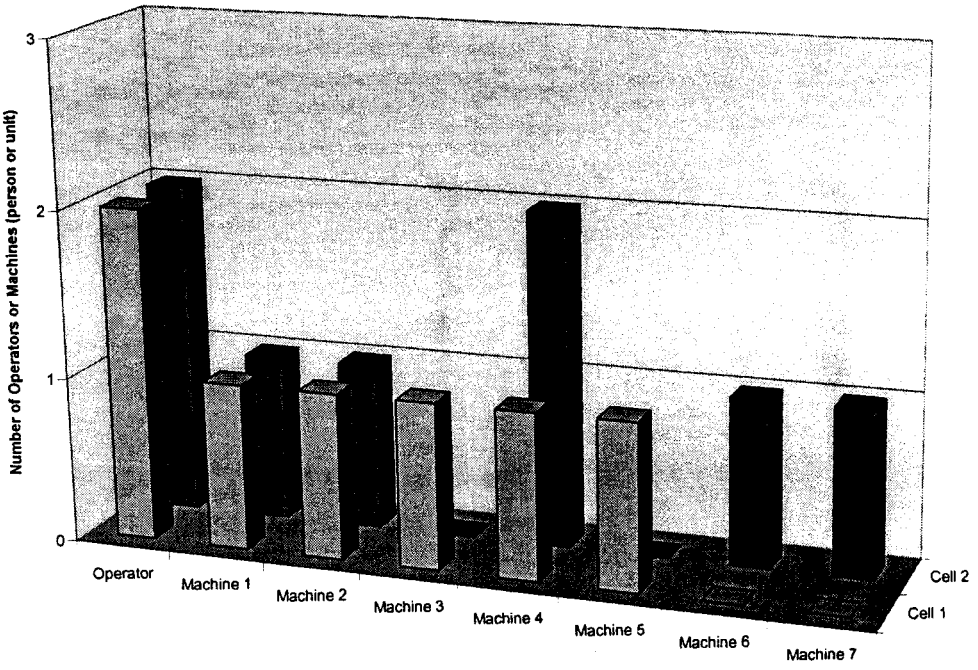

Figure 5.51: Operator and machine assignment in period 2

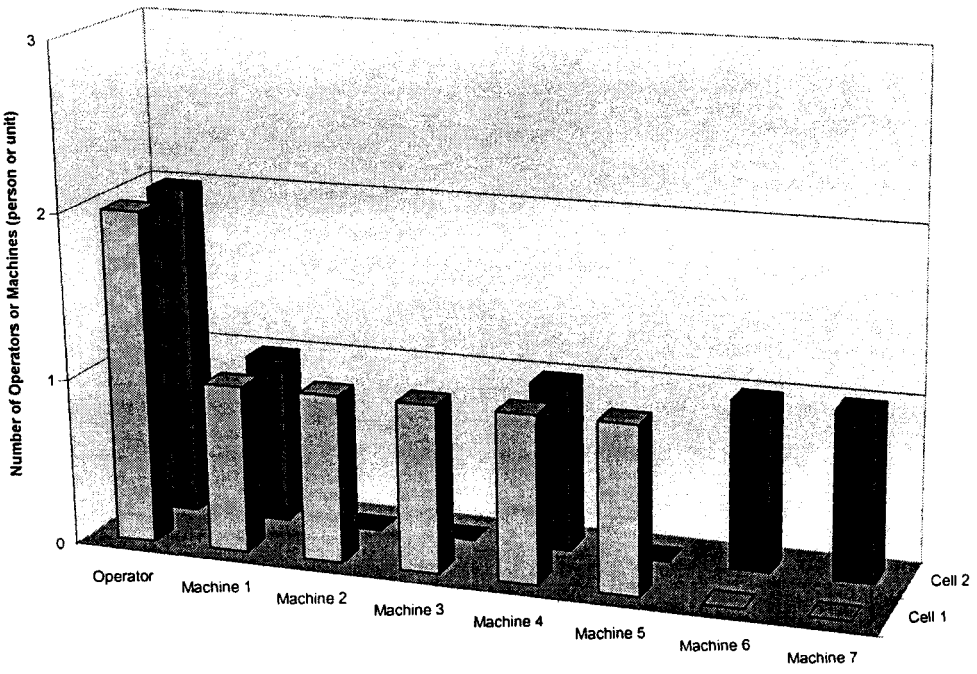

Figure 5.52: Operator and machine assignment in period 3

Table 5.69: Final cost values

Cost Function	Cost Value *($)*
Machine Capital Cost	674,000.10
Machine Idle Time Cost	108,946.70
Intercellular Movements Cost	29,599.98
Operator Cost	260,000.00
Operator Idle Time Cost	25,359.67
Lifting Risk Cost	324,165.40
Machine Relocation Cost	15,000.00
Manpower Level Change Cost	500.00
Total Cost	**1,437,571.85**

5.4.1.3. Result of SimGA Method (GA4)

Example 7 is solved simultaneously for the part families, machine cells formations, and operator assignment using GA4. The example is run for 3000 generations, with the population of 30 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 951st generation and the total time consumed is 19 minutes and 45 seconds (see Figure 5.53).

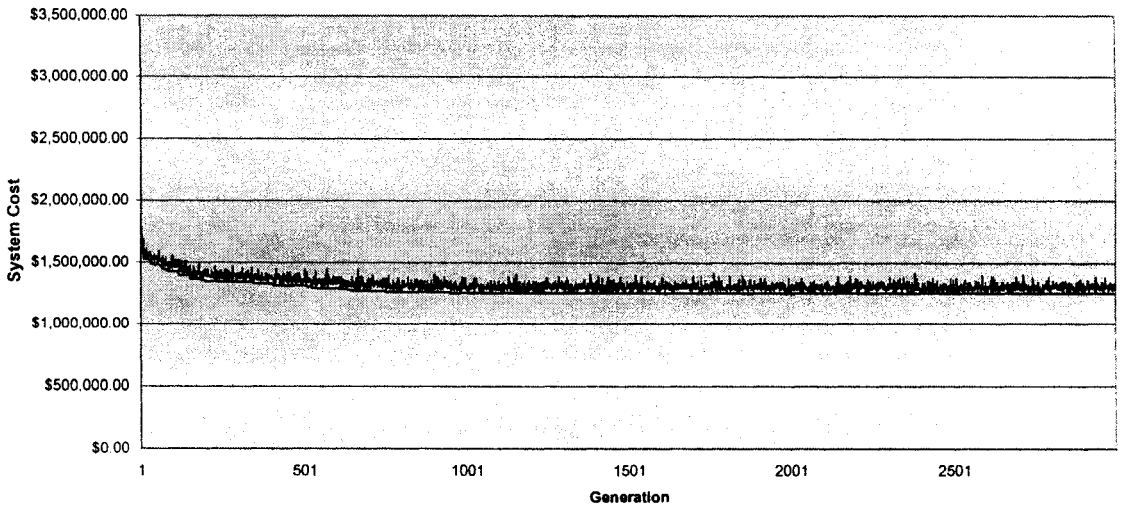

Figure 5.53: Average population cost and best cost for each generation

The results of part families and machine cells formations are summarized in Table 5.70. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.71. Also, various cost results of the solution are given in Table 5.72.

Table 5.70: Part families and machine cells formations

Period	Machine Cells	Parts Assigned	Machines Assigned *(unit)*							
			1	2	3	4	5	6	7	Total
1	Cell 1	1A, 1B, 2, 3, 4, 6A, 6B, 7, 8, 9, 10A	1	1	1	2	0	1	1	7
	Cell 2	1C, 5, 6C, 10B, 10C	0	1	0	1	1	0	0	3
2	Cell 1	1A, 1B, 2C, 4A, 7, 8, 9, 10A	1	0	1	2	0	1	0	5
	Cell 2	1C, 2A, 2B, 3, 4B, 4C, 5, 6, 10B, 10C	0	1	0	1	1	0	1	4
3	Cell 1	1, 2A, 3B, 3C, 4A, 5B, 5C, 6C, 7B, 7C, 8, 9, 10C	1	0	1	2	1	0	0	5
	Cell 2	2B, 2C, 3A, 4B, 4C, 5A, 6A, 6B, 7A, 10A, 10B	0	1	0	0	0	1	1	3

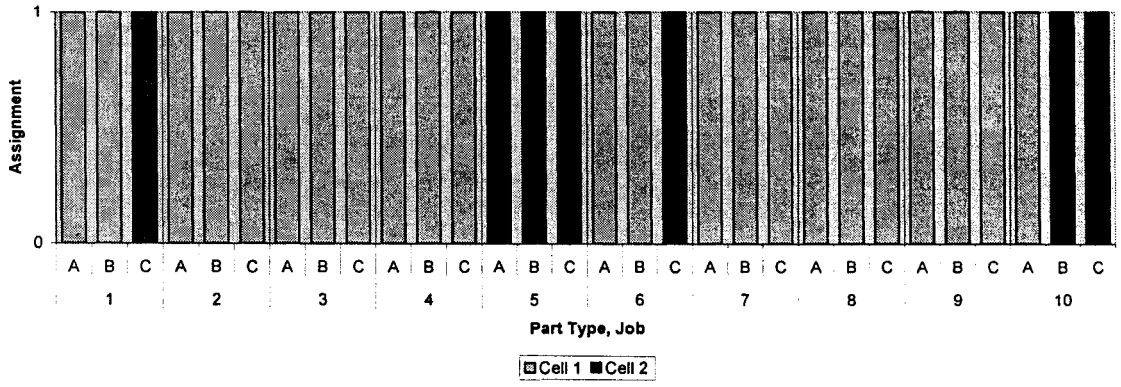

Figure 5.54: Part families in period 1

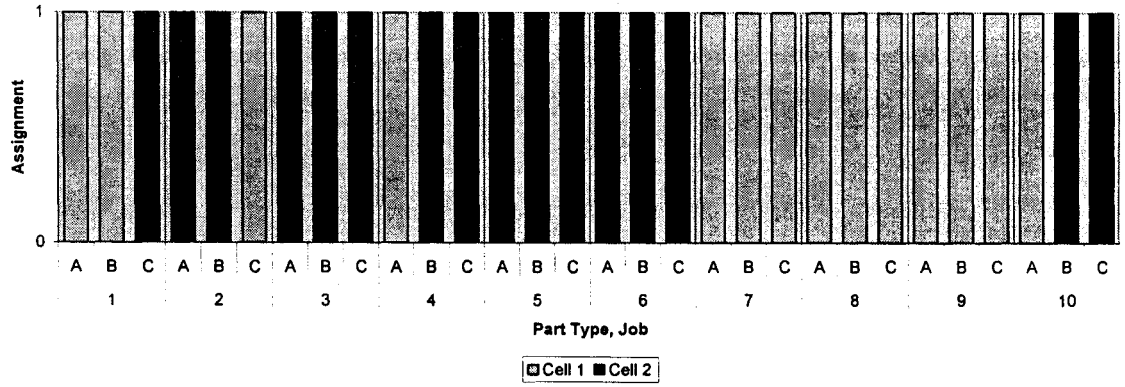

Figure 5.55: Part families in period 2

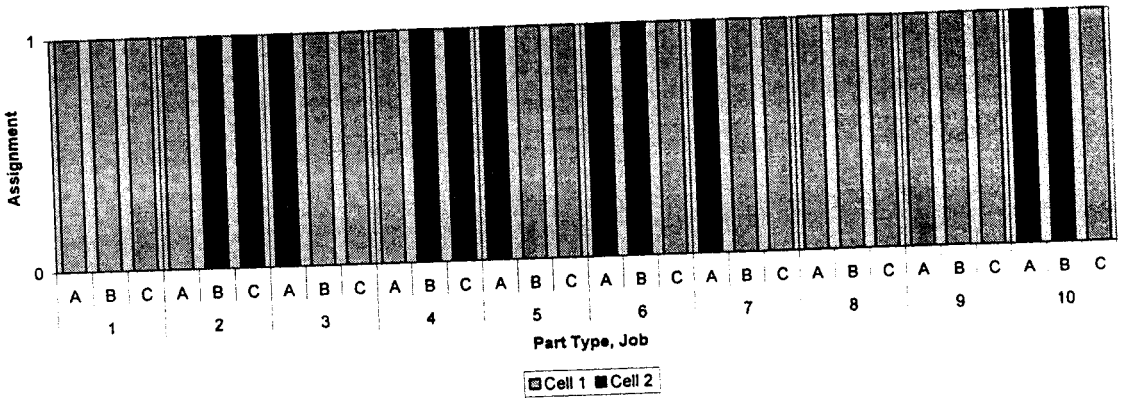

Figure 5.56: Part families in period 3

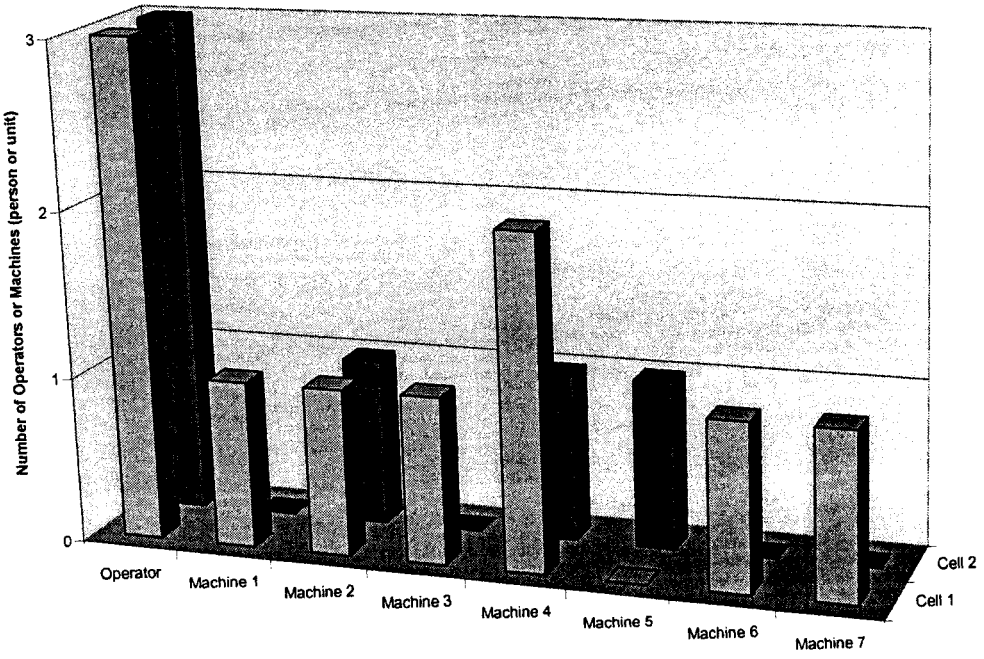

Figure 5.57: Operator and machine assignment in period 1

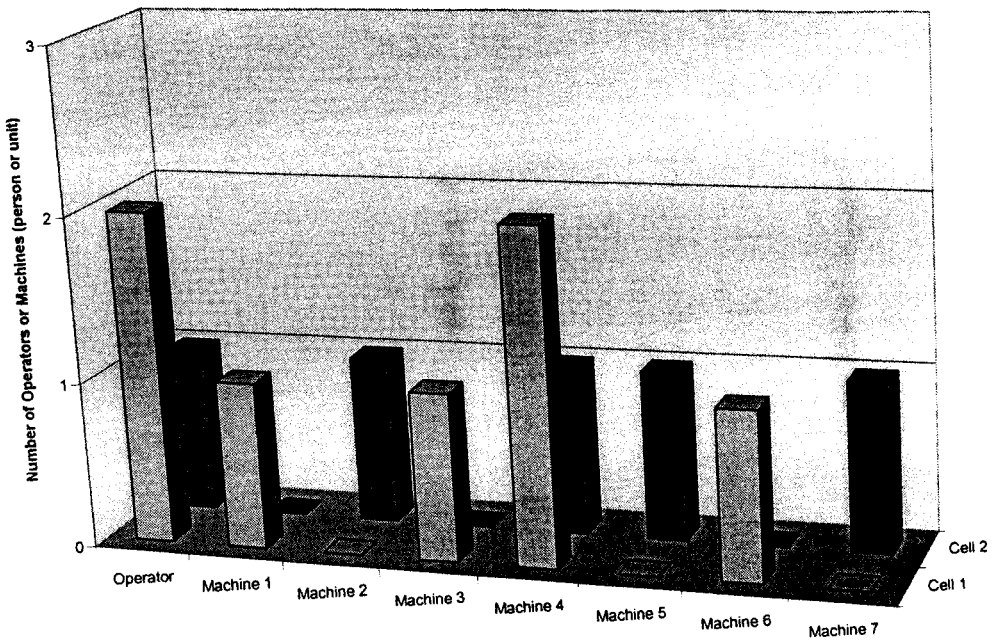

Figure 5.58: Operator and machine assignment in period 2

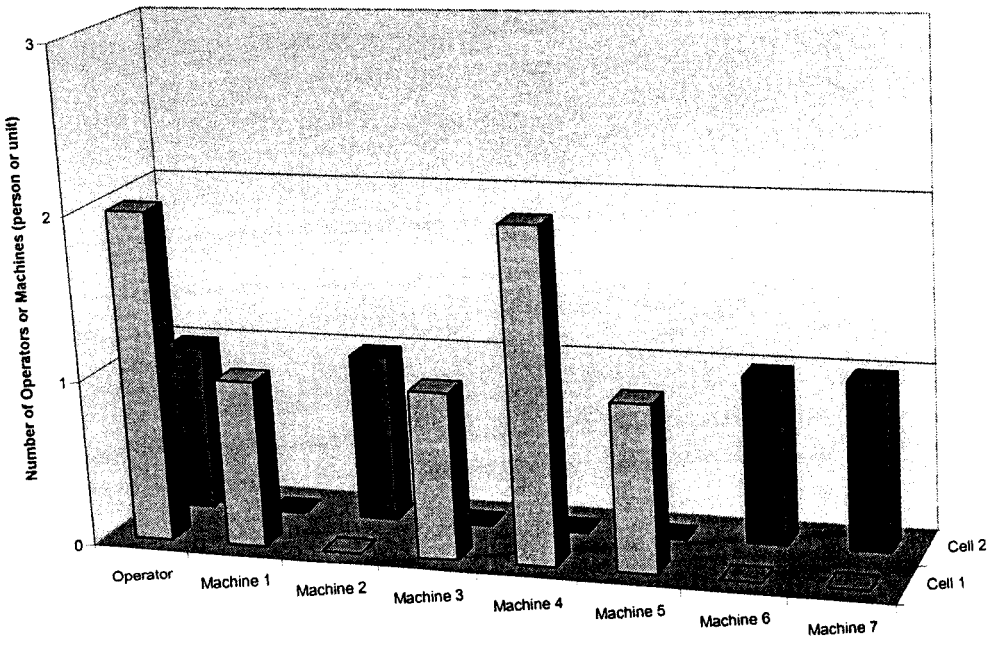

Figure 5.59: Operator and machine assignment in period 3

Table 5.71: Operator assignment, average frequency of lifting, and the composite lifting index

Period	Machine Cells	Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
1	Cell 1	3	1.008	1.238
	Cell 2	3	0.867	0.952
2	Cell 1	2	0.946	1.237
	Cell 2	1	1.067	1.301
3	Cell 1	2	0.842	1.267
	Cell 2	1	0.983	1.256

Table 5.72: Final cost values

Cost Function	Cost Value *($)*
Machine Capital Cost	578,000.00
Machine Idle Time Cost	62,946.67
Intercellular Movements Cost	84,150.00
Operator Cost	220,000.00
Operator Idle Time Cost	9,359.67
Lifting Risk Cost	263,646.09
Machine Relocation Cost	37,000.00
Manpower Level Change Cost	1,200.00
Total Cost	**1,256,302.43**

5.4.1.4. Comparison of Results of Example 7

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function values.

LINGO reported that Example 7 has:

- 1488 total variables, including 684 integer variables;
- 893 total constraints, including 379 nonlinear constraints; and
- 3450 total nonzeros, including 1314 nonlinear nonzeros.

The SeqBBH method cannot solve this example. This situation happens because inflexibility of solving the model sequentially. The SimBBH method is not the best in

terms of both running time and objective function. The SimGA method solved the problem with a reasonable running time and also gave the best system cost value.

So, if SimGA method compared with SeqBBH method, the reduction in computational time is 95.51% (26,397 seconds to 1,185 seconds), and the reduction in the system cost is 12.61% ($1,437,571.85 to $1,256,302.43). The detailed comparison of the three methods is given in Table 5.73.

Table 5.73: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: No Feasible Solution	Local Optimal	Best Solution found after 951st generation
Solver Time	15:43:19	07:19:57	00:19:45
Cost Elements:			
Machine Capital Cost	-	*674,000.10*	*578,000.00*
Machine Idle Time Cost	-	*108,946.70*	*62,946.67*
Intercellular Movements Cost	-	*29,599.98*	*84,150.00*
Operator Cost	-	*260,000.00*	*220,000.00*
Operator Idle Time Cost	-	*25,359.67*	*9,359.67*
Lifting Risk Cost	-	*324,165.40*	*263,646.09*
Machine Relocation Cost	-	*15,000.00*	*37,000.00*
Manpower Level Change Cost	-	*500.00*	*1,200.00*
Total System Cost	-	1,437,571.85	1,256,302.43

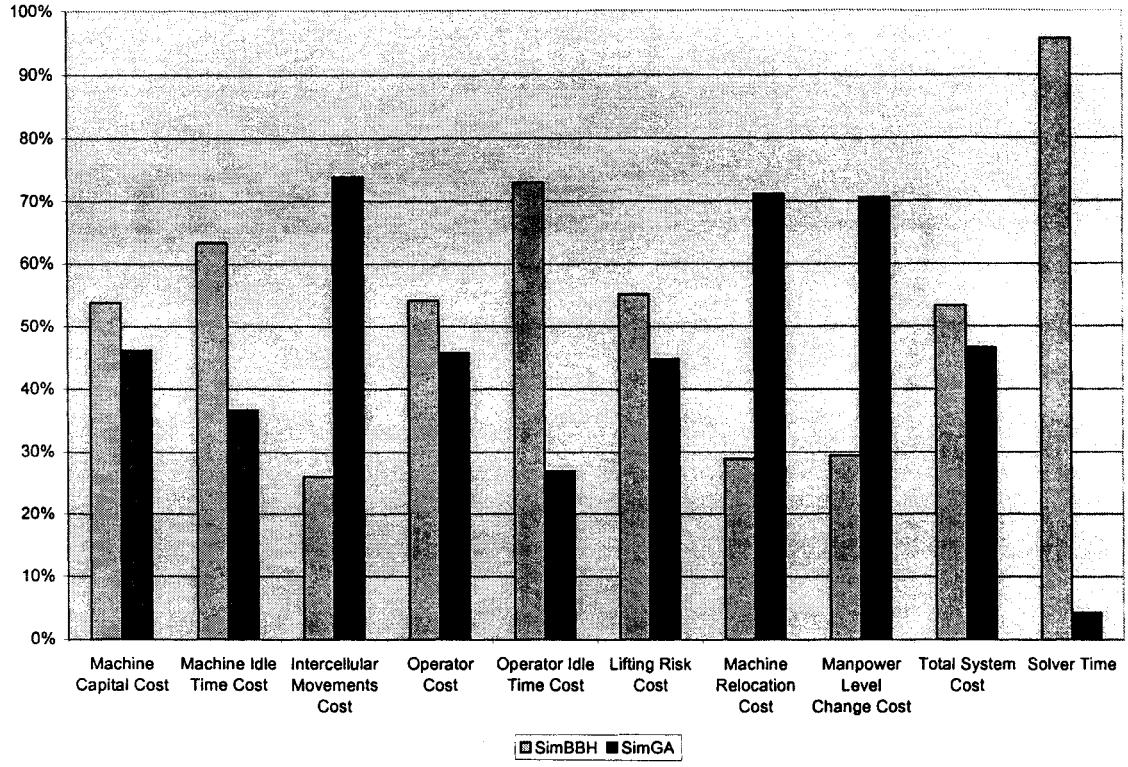

Figure 5.60: Costs and time comparisons

5.4.2. Example 8: Model 4 for Four-Cell Formation and Three-Period Planning

For this example 2000 machine hours capacity is assigned to each machine per period, the operator cost is \$10 per hour, operator idle time cost is \$4 per hour, and the approximate lifting risk compensation is \$20,000 per operator per lifting index.

The annual demands for each period, load weight, vertical manual lifting distance and intercellular movement cost for each part are given in Table 5.74. The machine requirements and the machine information are given in Table 5.75 and Table 5.76, respectively. The cell capacities for each period are given in Table 5.77. The machine relocation cost and the manpower level change cost are given in Table 5.78.

Table 5.74: Annual demands for each period, load weight, vertical lifting distance, and intercellular movement cost

Part Type	Demand *(unit)*				Load Weight *(kg)*	Vertical Lifting Distance *(cm)*	Intercellular Movement Cost *($)*
	Period 1	Period 2	Period 3	Period 4			
1	26000	22000	21000	19000	15	57	0.3
2	28000	16000	13000	9000	9	41	0.5
3	17000	18000	14000	8000	11	49	0.4
4	27000	21000	18000	17000	7	54	0.35
5	29000	23000	19000	15000	12	40	0.25
6	11000	9000	6000	9000	17	45	0.4
7	25000	19000	15000	19000	10	52	0.45
8	24000	20000	21000	20000	8	36	0.3
9	16000	20000	15000	20000	9	51	0.35
10	26000	9000	17000	19000	12	42	0.4
11	16000	13000	9000	15000	11	40	0.3
12	18000	15000	25000	16000	17	66	0.5
13	24000	10000	9000	12000	15	57	0.4
14	22000	16000	11000	8000	6	52	0.35
15	28000	14000	8000	16000	11	43	0.25
16	20000	22000	12000	12000	13	41	0.35
17	30000	24000	20000	28000	14	45	0.25
18	24000	20000	17000	12000	21	30	0.4
19	18000	19000	25000	17000	10	37	0.45
20	19000	22000	25000	20000	9	48	0.5

Table 5.75: Machine requirements

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 1	Job A	0	0	0	0	0	108	0	0	0	0	0	0
	Job B	0	0	114	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	126	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 2	Job A	0	72	0	0	0	0	0	0	0	0	180	0
	Job B	0	0	0	0	0	168	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	144	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 3	Job A	0	0	0	126	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	132	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 4	Job A	0	0	0	0	0	0	0	0	0	126	0	0
	Job B	0	126	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	138	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 5	Job A	0	72	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	0	0	114	0	0
	Job C	0	0	0	0	96	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 6	Job A	0	0	0	0	156	0	0	0	0	0	0	0
	Job B	174	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 7	Job A	0	90	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	84	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	192
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 8	Job A	0	0	0	0	0	0	0	132	0	0	0	0
	Job B	0	0	108	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	150	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 9	Job A	0	0	0	0	0	0	0	0	108	0	0	0
	Job B	0	0	0	0	0	0	168	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	102	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	126
Part 10	Job A	0	0	0	0	0	0	0	162	0	0	0	0
	Job B	90	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	138	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	72	0
Part 11	Job A	0	0	0	0	102	0	0	0	0	0	0	0
	Job B	0	0	0	72	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	108	0	0	0
	Job D	0	0	0	0	0	0	78	0	0	0	0	0
Part 12	Job A	0	126	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	96	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	78
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 13	Job A	0	0	0	0	0	0	0	0	0	0	90	0
	Job B	0	0	0	0	0	0	0	0	0	144	0	0
	Job C	0	0	126	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 14	Job A	0	0	0	0	0	0	0	0	0	0	0	126
	Job B	0	0	0	114	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0

Part Type & Jobs		Machining Time *(second)*											
		M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7	M/C 8	M/C 9	M/C 10	M/C 11	M/C 12
Part 15	Job A	0	0	150	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	150	0	0	0	0	0	0
	Job C	0	102	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 16	Job A	0	0	0	0	0	0	0	72	0	0	0	0
	Job B	138	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	78	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	192	0
Part 17	Job A	0	0	0	126	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	132	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	120	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 18	Job A	0	84	0	0	0	0	0	0	0	0	0	0
	Job B	0	0	0	0	0	0	0	132	0	0	0	0
	Job C	0	0	0	96	0	0	0	0	0	0	0	0
	Job D	78	0	0	0	0	0	0	0	0	0	0	0
Part 19	Job A	0	0	0	0	0	0	0	0	0	192	0	0
	Job B	102	0	0	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	0	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	0
Part 20	Job A	0	0	0	0	120	0	0	0	0	0	0	0
	Job B	0	0	90	0	0	0	0	0	0	0	0	0
	Job C	0	0	0	0	0	192	0	0	0	0	0	0
	Job D	0	0	0	0	0	0	0	0	0	0	0	108

Table 5.76: Capital cost, percentage of operator attention, and idle time cost of the machines

Machine Type	Capital Cost *($/period)*	Operator Attention Needed *(%)*	Idle Time Cost *($/hour)*
1	25000	0.75	4
2	26000	0.6	6
3	23000	0.75	5
4	15000	0.7	6
5	15000	0.65	3
6	24000	0.55	4
7	20000	0.75	4
8	18000	0.7	6
9	17000	0.5	5
10	16000	0.55	6
11	22000	0.5	5
12	21000	0.55	3

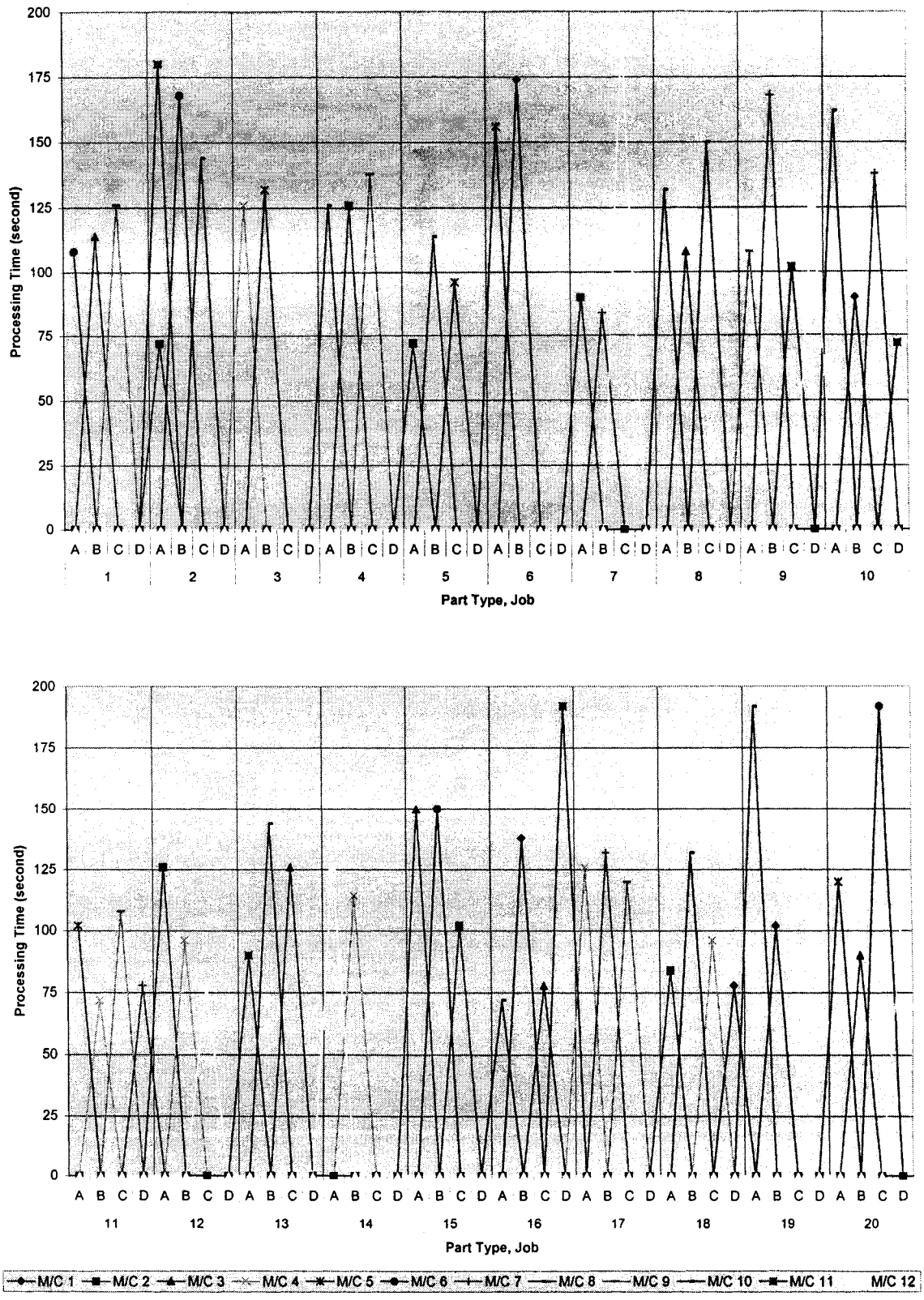

Figure 5.61: Processing time of each job on each part

Table 5.77: Maximum number of machines and operators allowed (cell capacity)

	Max. number of machines			Max. number of operators		
	Period 1	Period 2	Period 3	Period 1	Period 2	Period 3
Cell 1	12	12	11	8	8	7
Cell 2	14	14	12	8	8	7
Cell 3	12	11	9	7	7	7
Cell 4	14	14	14	8	7	8

Table 5.78: Machine relocation cost and manpower level change cost

	Period 1	Period 2	Period 3
Machine Increase	4000	6000	3000
Machine Decrease	5000	4000	5000
Operator Increase	600	400	700
Operator Decrease	600	500	300

5.4.2.1. Result of SeqBBH Method

Trials have been made to solve Example 8 sequentially: Step 1, part families and machine cells formations are solved in 25 hours, 36 minutes, and 51 seconds with global optimal state; and Step 2, operator assignment which is after 2 hours, 17 minutes, and 27 seconds, the LINGO solver could not find a feasible solution. This happens because after the Step 1, the part families and machine cells formations obtained make the model loses its flexibility to find a feasible solution on Step 2.

5.4.2.2. Result of SimBBH Method

Trials have been made to solve Example 8 simultaneously for the part families, machine cells formations, and operator assignment. However, after 5 hours, 48 minutes, and 1 second, the LINGO solver could not find a feasible solution. This happens because of the nonlinearity in the model.

5.4.2.3. Result of SimGA Method (GA4)

Example 8 is solved simultaneously for the part family, machine cell formation, and operator assignment using GA4. The example is run for 4000 generations, with the population of 40 chromosomes, probability of crossover (p_c) equal to 0.80, and probability of mutation (p_m) equal to 0.08. The best solution is obtained after 3858th generation and the total time consumed is 3 hours, 25 minutes and 28 seconds.

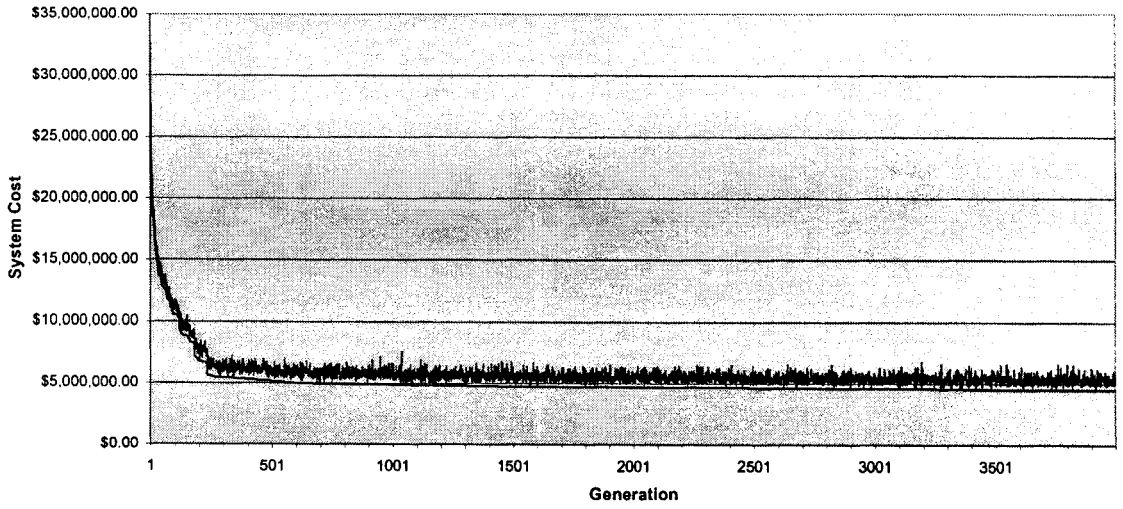

Figure 5.62: Average population cost and best cost for each generation

The results of part families and machine cells formations are summarized in Table 5.79 and Table 5.80, respectively. The number of operators assigned, average frequency of lifting, and the composite lifting index for each machine cell formation are summarized in Table 5.81. Also, various cost results of the solution are given in Table 5.82.

Table 5.79: Part families

		Parts Assigned
Period 1	Cell 1	1B, 5B, 6, 10D, 12B, 12C, 12D, 16B, 16C, 16D, 19A, 19B, 20A, 20B
	Cell 2	1A, 1C, 1D, 3A, 3C, 3D, 5D, 9B, 10A, 12A, 13, 14C, 14D, 16A, 17, 19C, 19D
	Cell 3	2A, 7A, 7C, 7D, 9C, 9D, 18A
	Cell 4	2B, 2C, 2D, 3B, 4, 5A, 5C, 7B, 8, 9A, 10B, 10C, 11, 14A, 14B, 15A, 15B, 18B, 18C, 18D, 20C, 20D
Period 2	Cell 1	3A, 3B, 6A, 6B, 10A, 10B, 11A, 12B, 16A, 17A, 19B
	Cell 2	1, 3C, 3D, 4A, 5B, 10D, 11C, 11D, 13A, 13B, 14C, 14D, 15B, 16B, 16C, 16D, 17B, 17C, 17D
	Cell 3	5A, 7A, 7C, 7D, 8A, 9B, 10C, 11B, 12A, 12C, 12D, 13C, 13D, 15A
	Cell 4	2, 4B, 4C, 4D, 5C, 5D, 6C, 6D, 7B, 8B, 8C, 8D, 9A, 9C, 9D, 14A, 14B, 15C, 15D, 18, 19A, 19C, 19D, 20
Period 3	Cell 1	5A, 6C, 6D, 7C, 7D, 8C, 9D, 10A, 10B, 10C, 12, 16A, 16B, 17B, 20A
	Cell 2	1A, 1B, 5B, 5C, 5D, 6A, 6B, 8B, 13, 15A, 15B, 16C, 16D, 19B, 19C, 19D
	Cell 3	1C, 1D, 11B, 11C, 17A, 17C, 17D
	Cell 4	2, 3, 4, 7A, 7B, 8A, 8B, 9A, 9B, 9C, 10D, 11A, 14, 15C, 15D, 18, 19A, 20B, 20C, 20D

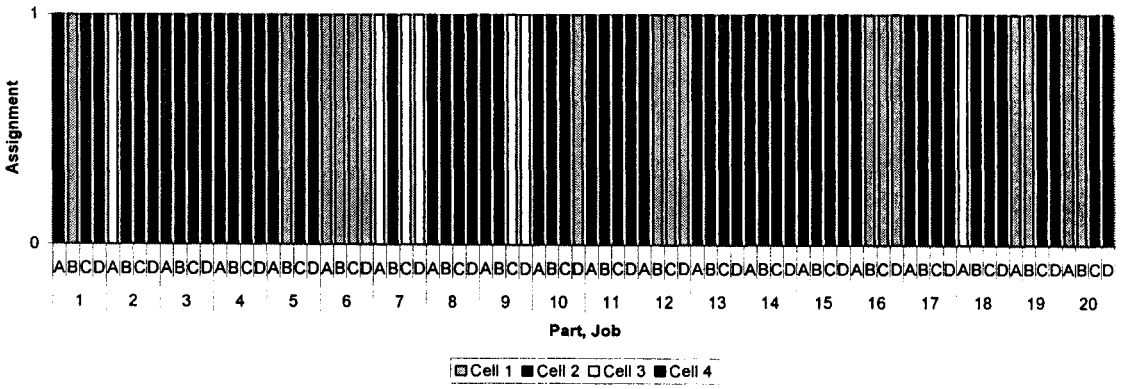

Figure 5.63: Part families in period 1

Figure 5.64: Part families in period 2

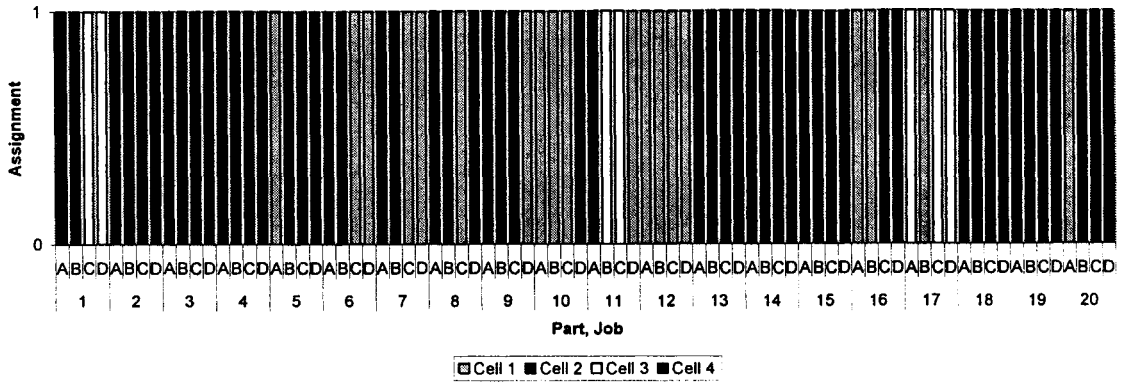

Figure 5.65: Part families in period 3

Table 5.80: Machine cells formations

		Machines Assigned *(unit)*												
		1	2	3	4	5	6	7	8	9	10	11	12	Total
Period 1	Cell 1	1	0	1	1	1	0	0	0	0	1	1	1	7
	Cell 2	0	1	1	1	0	1	1	1	1	1	1	0	9
	Cell 3	0	1	0	0	0	0	0	0	0	0	1	1	3
	Cell 4	1	1	1	1	1	2	1	2	1	1	0	1	13
Period 2	Cell 1	1	0	0	1	1	0	0	1	0	0	0	0	4
	Cell 2	1	0	1	0	0	1	1	0	1	1	1	0	7
	Cell 3	0	1	1	1	0	0	1	1	0	0	0	1	6
	Cell 4	1	1	1	1	1	1	1	1	1	1	1	1	12
Period 3	Cell 1	1	1	0	1	1	0	1	1	0	1	0	1	8
	Cell 2	1	0	1	0	1	1	0	0	0	1	1	0	6
	Cell 3	0	0	0	1	0	0	0	0	1	0	0	0	2
	Cell 4	1	1	1	1	1	1	1	1	1	1	1	1	12

Table 5.81: Operator assignment, average frequency of lifting, and the composite lifting index

		Number of Operators	Average Frequency of Lifting *(lifts/min)*	Composite Lifting Index
Period 1	Cell 1	3	0.758	1.392
	Cell 2	4	0.706	1.401
	Cell 3	2	0.558	1.498
	Cell 4	7	0.775	1.499
Period 2	Cell 1	2	0.688	1.393
	Cell 2	3	0.814	1.241
	Cell 3	2	0.738	1.373
	Cell 4	5	0.775	1.499
Period 3	Cell 1	3	0.761	1.382
	Cell 2	2	0.767	1.359
	Cell 3	1	0.658	1.250
	Cell 4	5	0.770	1.499

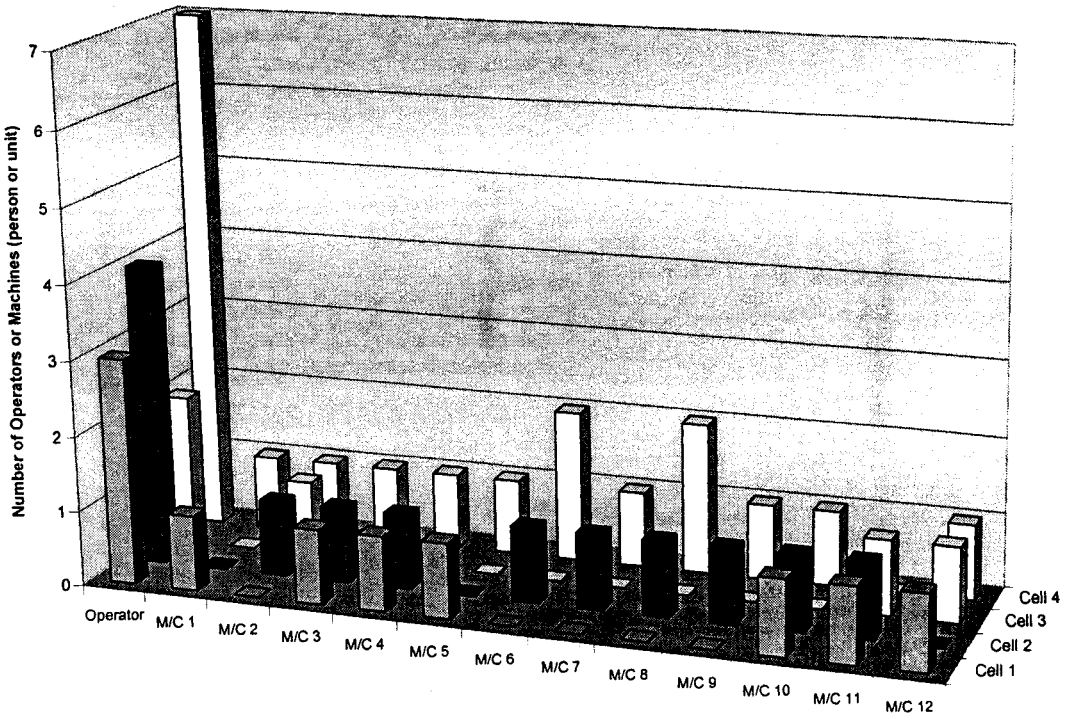

Figure 5.66: Operator and machine assignment in period 1

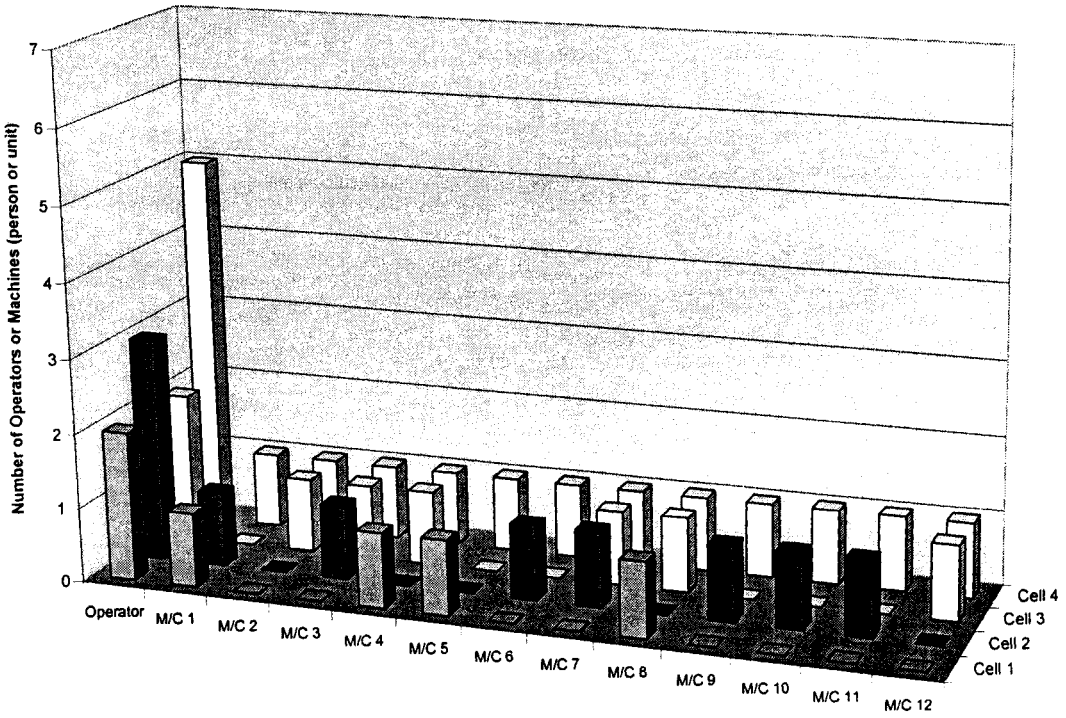

Figure 5.67: Operator and machine assignment in period 2

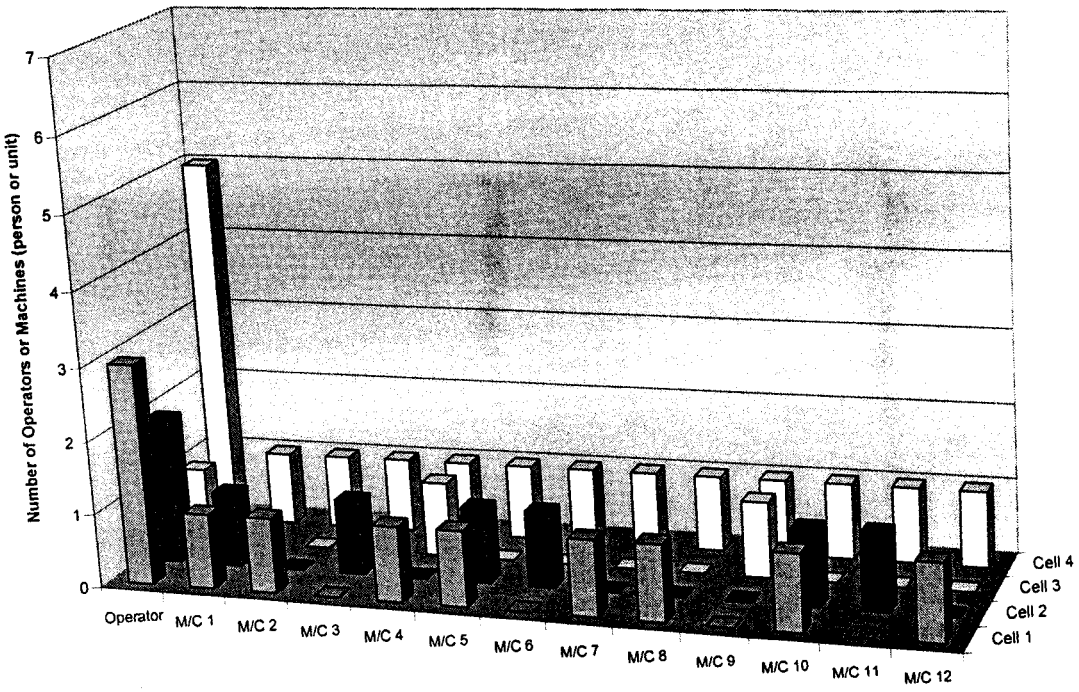

Figure 5.68: Operator and machine assignment in period 3

Table 5.82: Final cost values

Cost Function	Cost Value ($)
Machine Capital Cost	1,789,000.00
Machine Idle Time Cost	279,485.00
Intercellular Movements Cost	411,700.00
Operator Cost	780,000.00
Operator Idle Time Cost	14,863.00
Lifting Risk Cost	1,112,726.61
Machine Relocation Cost	135,000.00
Manpower Level Change Cost	3,300.00
Total Cost	**4,526,074.61**

5.4.2.4. Comparison of Results of Example 8

The results from the three methods: SeqBBH, SimBBH, and SimGA; are compared and analyzed in terms of computational time and objective function values.

LINGO reported that Example 8 has:

- 7408 total variables, including 3348 integer variables;
- 3857 total constraints, including 1957 nonlinear constraints; and
- 17052 total nonzeros, including 6852 nonlinear nonzeros.

The detailed comparison of the three methods is given in Table 5.83.

Table 5.83: Comparison of the three methods

Method	SeqBBH	SimBBH	SimGA
Model Type	Linear & Nonlinear	Nonlinear	Nonlinear
Solution Type	Step 1: Global Optimal Step 2: No Feasible Solution	No Feasible Solution	Best Solution found after 3675th generation
Solver Time	27:54:18	05:48:01	03:25:28
Cost Elements:			
Machine Capital Cost	-	-	*1,789,000.00*
Machine Idle Time Cost	-	-	*279,485.00*
Intercellular Movements Cost	-	-	*411,700.00*
Operator Cost	-	-	*780,000.00*
Operator Idle Time Cost	-	-	*14,863.00*
Lifting Risk Cost	-	-	*1,112,726.61*
Machine Relocation Cost	-	-	*135,000.00*
Manpower Level Change Cost	-	-	*3,300.00*
Total System Cost	-	-	4,526,074.61

5.5. Consistency of Genetic Algorithms

In this section, the consistency of Genetic Algorithms (GA1, GA2, GA3, and GA4) is tested by running the algorithms for 10 times using 8 examples presented before. Then the average and standard deviation of the system costs resulted are calculated. The objective is to observe the algorithms' accuracy in finding the solution each time they are run by calculating the ratio between the standard deviation and the average system costs. The computational times to solve the examples are also compared in the same method.

The ratio between the standard deviation and the average systems costs from the calculation are within 2%, which means there are less than 2% of errors from the average system cost. The systems cost for each trial and the ratio for each example are presented in Table 5.84.

The ratio between the standard deviation and the average time consumptions from the calculation are within 3%, which means there are less than 3% of errors from the average computational time. The computational time for each trial and the ratio for each example are presented in Table 5.85.

Table 5.84: The system costs and the error ratios

Trial	System Cost *($)*							
	Example 1	Example 2	Example 3	Example 4	Example 5	Example 6	Example 7	Example 8
#1	431,771.73	840,767.70	699,839.79	1,514,867.06	1,321,376.74	4,984,686.24	1,306,672.92	4,796,361.64
#2	431,771.73	883,628.09	722,028.37	1,485,662.77	1,369,778.61	5,198,276.73	1,259,102.99	4,591,855.56
#3	431,771.73	844,887.30	694,815.81	1,512,283.82	1,355,276.76	5,071,918.83	1,271,591.60	4,526,074.61
#4	431,771.73	850,962.53	705,319.05	1,482,382.08	1,321,376.74	5,247,218.76	1,269,378.07	4,559,475.63
#5	431,771.73	844,887.30	689,801.49	1,484,992.51	1,355,276.76	5,287,609.49	1,259,102.99	4,544,996.06
#6	431,771.73	882,629.52	705,319.05	1,500,572.76	1,321,376.76	5,200,807.69	1,256,302.42	4,716,239.68
#7	431,771.73	844,887.30	708,984.40	1,543,591.61	1,366,116.16	5,198,481.01	1,303,670.89	4,687,361.35
#8	431,771.73	845,753.50	721,363.34	1,542,742.02	1,355,276.76	5,275,485.25	1,256,821.28	4,602,951.78
#9	431,771.73	840,767.72	711,467.93	1,518,374.75	1,369,778.61	5,247,218.76	1,261,980.87	4,660,850.91
#10	431,771.73	844,887.30	735,576.18	1,540,175.43	1,321,376.76	5,071,918.83	1,282,833.80	4,594,401.78
Average	***431,771.73***	***852,405.83***	***709,451.54***	***1,512,564.48***	***1,345,701.07***	***5,178,362.16***	***1,272,745.78***	***4,628,056.90***
Std Dev	*0.00*	*16,437.23*	*13,799.03*	*24,068.34*	*21,648.72*	*101,284.10*	*18,955.98*	*85,486.52*
Ratio%	*0.00%*	*1.93%*	*1.95%*	*1.59%*	*1.61%*	*1.96%*	*1.49%*	*1.85%*

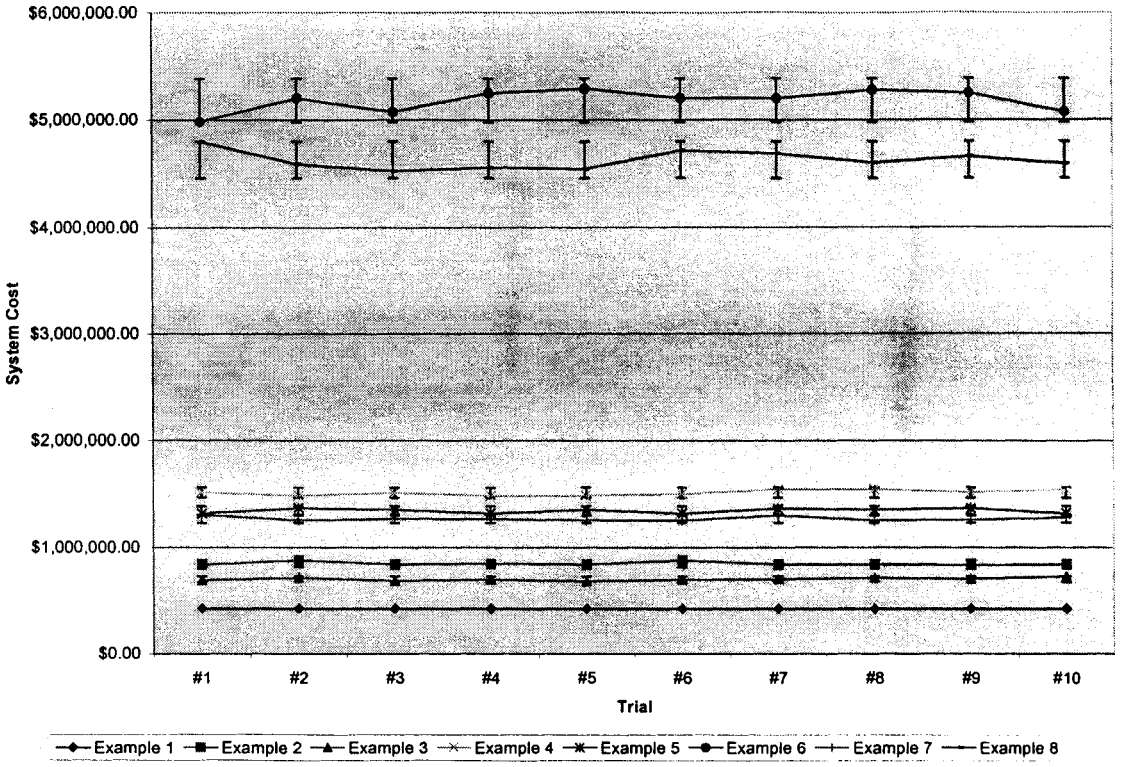

Figure 5.69: System costs and 2% error bars

Table 5.85: The computational time and the error ratios

Trial	Computational Time *(second)*							
	Example 1	Example 2	Example 3	Example 4	Example 5	Example 6	Example 7	Example 8
#1	52	315	144	2,010	363	2,378	1,203	12,268
#2	53	328	143	1,997	358	2,298	1,185	12,218
#3	53	319	142	1,977	360	2,269	1,171	12,328
#4	55	320	145	1,935	383	2,343	1,114	11,822
#5	56	316	148	1,986	367	2,305	1,135	11,942
#6	53	317	146	1,989	380	2,296	1,185	12,447
#7	54	322	147	1,975	375	2,275	1,146	12,321
#8	55	320	149	2,058	368	2,308	1,131	12,364
#9	52	319	143	2,098	371	2,318	1,128	12,542
#10	53	319	142	2,008	381	2,287	1,133	12,721
Average	*54*	*320*	*145*	*2,003*	*371*	*2,308*	*1,153*	*12,297*
Std Dev	*1.350*	*3.629*	*2.514*	*45.617*	*8.934*	*32.537*	*30.322*	*263.957*
Ratio%	*2.52%*	*1.14%*	*1.74%*	*2.28%*	*2.41%*	*1.41%*	*2.63%*	*2.15%*

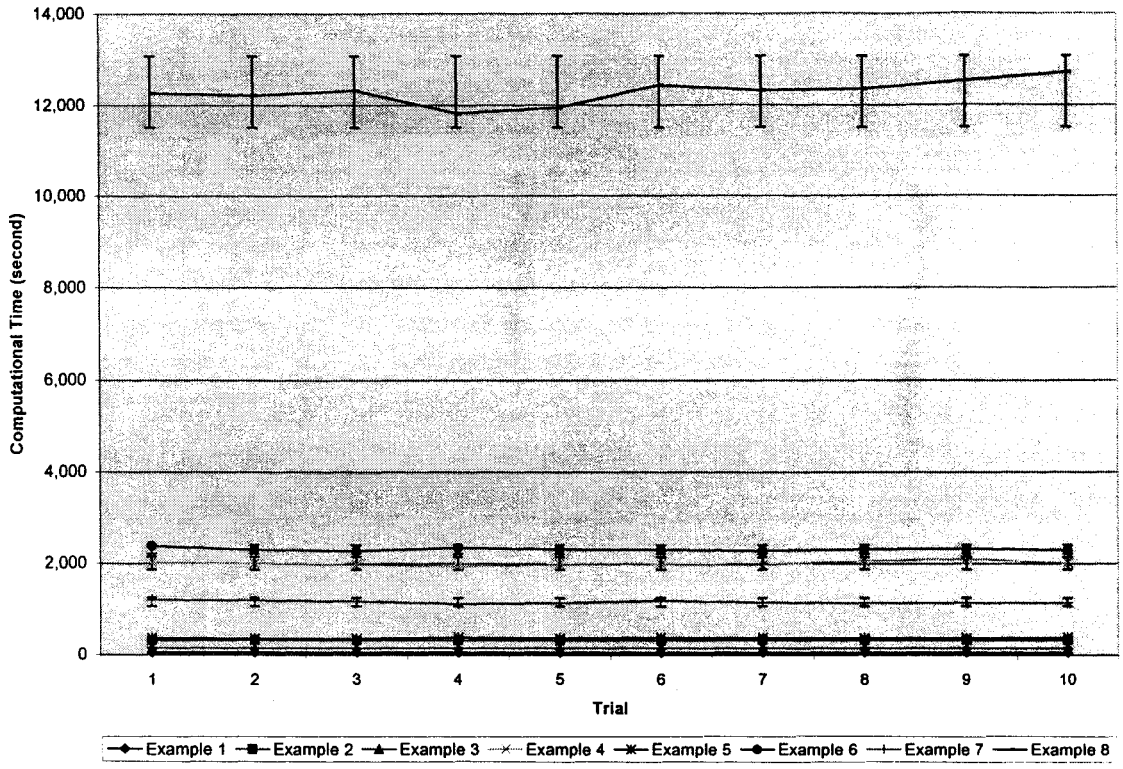

Figure 5.70: Computational times and 3% error bars

Chapter 6 –
Conclusions and Future Research

6.1. Summary and Conclusions

Based on the literature review, previous research in cellular manufacturing systems or other manufacturing systems tend to ignore or forget the human factors or human issues involved in the manufacturing system. In this research, mathematical models are developed to simultaneously solve the part family, machine cell formation, and operator assignment problems under single and multi period planning horizons. The objective of these models is to minimize the system costs while attempting to develop independent machine cells and ensure that the operators involved in the manual-material-handling activities are working in an ergonomically safe condition based on the NIOSH Lifting Guidelines.

The single period models (Model 1 and Model 2) consider the economic cost trade-offs between machine duplication, operator cost, machine idle time, operator idle time, intercellular movements (Model 2 only) and lifting risk penalty. Lifting risk penalty in the models is based on the revised NIOSH lifting guidelines. Several adaptations in the lifting equation have been made to incorporate the lifting equation into the models.

The multi period models (Model 3 and Model 4) consider the same cost elements in the single period models with addition of machine relocation and manpower level change for each period in the planning horizon.

A heuristic technique, based on the mathematical models, is developed using Genetic Algorithms. The objectives of developing this heuristic technique are: to deal with the nonlinearity in the models that makes some difficulties to find global optimal solution, and to reduce the amount of computational time needed to solve large size problems. The algorithms also consider the same objective function and constraints as specified in the mathematical models.

The models are tested with different sizes of numerical examples in three methods: sequentially using combination of Branch-and-Bound and Heuristic (SeqBBH), simultaneously using combination of Branch-and-Bound and Heuristic (SimBBH), and simultaneously using Genetic Algorithm (SimGA). The solutions give the part families (parts assignment to the cells), machine groups (machines assignment to the cells), and operator assignment based on NIOSH lifting guidelines.

Single period model with job sequence (Model 2) creates intercellular movements instead of duplicating the bottleneck machines based on the cost trade-off. Multi period models create a balanced machine and operator capacity distribution through the planning periods by changing the configuration of machine and operator in each period. It is concluded that all mathematical models achieved independent machine cell formations based on economic trade-offs.

The SimGAs (GA1, GA2, GA3, and GA4) developed are tested and compared with the SeqBBH and SimBBH in terms of objective function values and computational times to solve the examples. From Table 6.1, it is obvious that the heuristic method gave better solution both in system cost value and computational time. It is concluded that the

ability of the SeqBBH and SimBBH to solve the examples is not effective and efficient because of nonlinearity in the models (see Table 6.1 for details).

Table 6.1: Comparison of solver solutions

Model	Method	SeqBBH	SimBBH	SimGA
Model 1: Single Period	Example 1	$442,828.33	$431,771.73	$431,771.73
	Example 2	$864,569.90	No Solution	$840,767.70
Model 2: Single Period	Example 3	$779,102.46	No Solution	$689,801.49
	Example 4	No Solution	No Solution	$1,482,382.09
Model 3: Multi Period	Example 5	$1,321,376.74	No Solution	$1,321,376.74
	Example 6	No Solution	No Solution	$4,984,686.24
Model 4: Multi Period	Example 7	No Solution	$1,437,571.85	$1,256,302.43
	Example 8	No Solution	No Solution	$4,526,074.61

Finally, the consistency of GAs is tested by performing 10 runs on the same examples to find the deviations or errors. The objective is to observe the algorithms' accuracy in finding the solution each time they are run by calculating the ratio between the standard deviation and the average system costs. The computational times to solve the examples are also compared in the same method. The ratio of the standard deviation and the average systems costs from the calculation are within 2%, which means there are less than 2% of errors from the average system cost. The ratio between the standard deviation and the average time consumptions from the calculation are within 3%, which means there are less than 3% of errors from the average computational time.

From the comparisons of examples and the consistency tests, it is concluded that solving part family, machine cell formation, and operator assignment simultaneously is better than to solve them sequentially. It is also concluded that the Genetic Algorithms

developed in Chapter 4 are reliable and consistent to solve the mathematical models with a reasonable computational time.

6.2. Future Research Recommendation

In the mathematical models, limited cost functions are considered. More cost factors, such as tooling cost and intracellular movements cost, can be incorporate into the model for further detail modeling. Alternative or multiple process plans for each part type can be developed in the model instead of one process plan. Additionally, subcontracting of part is also can be included in the model.

The human factors element in the manual-material-handling activities in this model is only limited to the manual lifting tasks, and also limited to the revised NIOSH lifting guidelines. In practices, the manual-material-handling activities include: lifting, lowering, carrying, pulling, pushing, or even over reach lifting/lowering. These activities should be included further to give better and more accurate models in operator assignment considering human factors or ergonomics.

Other approaches to design the manual-material-handling activities can also be developed, for example, based on psychophysical approach, physiological approach, or epidemiological approach.

Finally, further considerations of ergonomic areas related to the manufacturing systems are needed to create better working environment in manufacturing cells.

References

Adenzo-Díaz, B., Lozano, S., Racero, J., and Guerrero, F. (2001), "Machine cell formation in generalized group technology", *Computers and Industrial Engineering*, **41**, pp.227-247.

Alfa, A.S., Chen, M., and Heragu, S. (1992), "Integrating the grouping and layout problems in cellular manufacturing systems", *Computers and Industrial Engineering*, **23**, pp.55-58.

Anderberg, M.R. (1973), *Cluster analysis for applications*, New York: Academic Press.

Askin, R.G., and Chiu, K. S. (1990), "A graph partitioning procedure for machine assignment and cell formation in group technology", *International Journal of Production Research*, **28**, pp.1555-1572.

Askin, R.G., and Estrada, S. (1999), "Investigation of cellular manufacturing practices", *Handbook of Cellular Manufacturing Systems*, ed. S.A. Irani, John Wiley & Sons, New York, pp.25-34.

Askin, R.G., and Huang, Y. (1997), "Employee training and assignment for facility configuration", *Proceedings of 6th Industrial Engineering Research Conference*, pp.426-431.

Askin, R.G., and Subramanian, S.P. (1987), "A cost-based heuristic for group technology configuration", *International Journal of Production Research*, **25**, pp.101-113.

Askin, R.G., Cresswell, S.H., Goldberg, J.B., and Vakharia, A.J. (1991), "A Hamiltonian path approach to reordering the part-machine matrix for cellular manufacturing", *International Journal of Production Research*, **29**, pp.1081-1100.

Askin, R.G., Selim, H.M., and Vakharia, A.J. (1997), "A methodology for designing flexible cellular manufacturing systems", *IEE Transactions*, **29**, pp.599-610.

Balasubramanian, K.N., and Panneerselvam, R. (1993), "Covering technique-based algorithm for machine grouping to form manufacturing cells", *International Journal of Production Research*, **31**, pp.1479-1504.

Ballakur, A., and Steudel, H. J. (1987), "A within-cell utilization based heuristic for designing cellular manufacturing systems", *International Journal of Production Research*, **25**, pp.639-665.

Baykasoğlu, A., Gindy, N.N.Z., and Cobb, R.C. (2001), "Capability based formulation and solution of multiple objective cell formation problems using simulated annealing", *Integrated Manufacturing Systems*, **12**, pp.258-274.

Beaulieu, A., Gharbi, A., and Air-Kadi, D. (1997), "An algorithm for the cell formation and the machine selection problems in the design of a cellular manufacturing system", *International Journal of Production Research*, **35**, pp.1857-1874.

Ben-Arieh, D. (1998), "Analysis of a distributed group technology methodology", *Computers and Industrial Engineering*, **35**, pp.69-72.

Ben-Arieh, D., and Sreenivasan, R. (1999), "Information analysis in a distributed dynamic group technology method", *International Journal of Production Economics*, **60-61**, pp.427-432.

Black, J.T. (1991), *The design of the factory with a future*, McGraw-Hill, Reading.

Burke, L, and Kamal, S. (1995), "Neural networks and the part family / machine group formation problem in cellular manufacturing: a framework using fuzzy ART", *Journal of Manufacturing Systems*, **14**, pp.148.

Burbidge, J.L. (1963), "Production flow analysis", *The Production Engineer*, **42**, pp.742.

Carrie, A.S. (1973), "Numerical taxonomy applied to group technology and plant layout", *International Journal of Production Research*, **11**, pp.399-416.

Chan, H.M., and Milner, D.A. (1982), "Direct clustering algorithm for group formation in cellular manufacturing", *Journal of Manufacturing Systems*, **1**, pp.65-74.

Chandrasekharan, M.P., and Rajagopalan, R. (1986a), "An ideal seed non-hierarchical clustering algorithm for cellular manufacturing", *International Journal of Production Research*, **24**, pp.451-464.

Chandrasekharan, M.P., and Rajagopalan, R. (1986b), "MODROC: an extension of rank order clustering for group technology", *International Journal of Production Research*, **24**, pp.1221-1233.

Chandrasekharan, M.P., and Rajagopalan, R. (1987), "ZODIAC – an algorithm for concurrent formation of part-families and machine-cells", *International Journal of Production Research*, **25**, pp.835-850.

Chen, H.G. (1995), "Operator scheduling approaches in group technology cells – information request analysis", *IEEE Transactions on Systems, Man, and Cybernetics*, 25, pp.438-440.

Chen, J.S. (1995), *Cellular manufacturing system design: optimal solution of large scale real world problems*, PhD thesis, Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, Troy, New York.

Cheng, C.H., Goh, C.H., and Lee, A. (2001), "Designing group technology manufacturing systems using heuristics branching rules", *Computers and Industrial Engineering*, **40**, pp.117-131.

Choobineh, F. (1988), "A framework for the design of cellular manufacturing systems", *International Journal of Production Research*, **26**, pp.1161-1172.

Chow, W.S., and Hawaleshka, O. (1993), "Minimizing intercellular part movements in manufacturing cell formation", *International Journal of Production Research*, **31**, pp.2161-2170.

Chu, C.H. (1993), "Manufacturing cell formation by competitive learning", *International Journal of Production Research*, **31**, pp.829-843.

Chu, C.H., and Tsai, M. (1990), "A comparison of three array-based clustering techniques for manufacturing cell formation", *International Journal of Production Research*, **28**, pp.1417-1433.

Chung, M.K., and Kee, D. (2000), "Evaluation of lifting tasks frequently performed during fire brick manufacturing processes using NIOSH lifting equations", *International Journal of Industrial Ergonomics*, **25**, pp.423-433.

Chung, Y., and Kusiak, A. (1994), "Grouping parts with a neural network", *Journal of Manufacturing Systems*, **13**, pp.262.

Dahel, N.E., and Smith, S.B. (1993), "Designing flexibility into cellular manufacturing systems", *International Journal of Production Research*, **31**, pp.933-945.

de Witte, J. (1980), "The use of similarity coefficients in production flow analysis", *International Journal of Production Research*, **18**, pp.503-514.

Diallo, M., Pierreval, H., and Quilliot, A. (2001), "Manufacturing cells design with flexible routing capability in presence of unreliable machines", *International Journal of Production Economics*, **74**, pp.175-182.

Eckstein, A.L.H., and Rohleder, T.R., (1998), "Incorporating human resources in group technology/cellular manufacturing", *International Journal of Production Research*, **36**, pp.1199-1222.

El-Essawy, I.F.K., and Torrance, J. (1972), "Component flow analysis – an effective approach to production systems' design", *The Production Engineer*, **51**, pp.165.

Elmaraghy, H. A., and Gu, P. (1988), "Feature based expert parts assignment in cellular manufacturing", *Journal of Manufacturing Systems*, **8**, pp.139-152.

Enke, D., Ratanapan, K., and Dagli, C. (1998), "Machine-part family formation utilizing an ART1 neural network implemented on a parallel neuro-computer", *Computers and Industrial Engineering*, **34**, pp.189-205.

Faber, Z., and Carter, M. W. (1986), "A new graph theory approach for forming machine cells in cellular production systems", *Flexible Manufacturing Systems: Methods and Studies*, ed. A. Kusiak, North-Holland, New York, pp.301-318.

Fan, I.S., and Gassmann, R. (1995), "Study of practicalities of human centered implementation in a British manufacturing company", *Computer Integrated Manufacturing Systems*, **8**, pp.151-154.

Gen M., and Cheng, R. (1997), *Genetic algorithms and engineering design*, John Wiley & Sons, New York.

Goldberg, D. (1989), *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Publishing Company, Don Mills, Ontario.

Gupta, T. (1993), "Design of manufacturing cells for flexible environment considering alternative routeing", *International Journal of Production Research*, **31**, pp.1259-1273.

Gupta, T., and Seifoddini, H. (1990), "Production data based similarity coefficient for machine-component grouping decisions in the design of a cellular manufacturing system", *International Journal of Production Research*, **28**, pp.1247-1269.

Gwiazda, A., and Knosala, R. (1997), "Group technology using neural nets", *Journals of Materials Processing Technology*, **64**, pp.181-188.

Houtzeel, A., and Brown, C.S. (1984), "A management overview of group technology", *Group Technology at Work*, ed. N.L. Hyer, Society of Manufacturing Engineers, Detroit, Michigan, pp.3-16.

Huber, V.L., and Brown, K.A. (1991), "Human resource issues in cellular manufacturing: a sociotechnical analysis", *Journal of Operations Management*, **10**, pp.138-159.

Irani, S.A., Subramanian, S., and Allam, Y.S. (1999), "Introduction to Cellular Manufacturing Systems", *Handbook of Cellular Manufacturing Systems,* ed. S.A. Irani, John Wiley & Sons, New York.

Jain, A.K., Kasilingam, R.G., and Bhole, S.D. (1990), "Cell formation in flexible manufacturing systems under resource constraints", *Computers and Industrial Engineering*, **19**, pp.437-441.

Kamrani, A.K., Hubbard, K., Parsaei, H.R., and Leep, H.R. (1998), "Simulation-based methodology for machine cell design", *Computers and Industrial Engineering*, **34**, pp.173-188.

Kandiller, L. (1998), "A cell formation algorithm: Hypergraph approximation – Cut tree", *European Journal of Production Research*, **109**, pp.686-702.

Kaparthi, S., and Suresh, N. C. (1991), "A neural network system for shape-based classification and coding of rotational parts", *International Journal of Production Research*, **29**, pp.1771-1784.

Kaparthi, S., and Suresh, N. C. (1992), "Machine-component cell formation in group technology: a neural network approach", *International Journal of Production Research*, **30**, pp.1353-1367.

Kern, G.M., and Wei, J.C. (1991), "The cost of eliminating exceptional elements in group technology cell formation", *International Journal of Production Research*, **29**, pp.1535-1547.

Khator, S.K., and Irani, S.A. (1987), "Cell formation in group technology: a new approach", *Computers and Industrial Engineering*, **12**, pp.131-142.

Kiang, M.Y., Kulkarni, U.R., and Tam, K.Y. (1995), "Self-organizing map network as an interactive clustering tool – an application to group technology", *Decision Support Systems*, **15**, pp.351.

King, J.R. (1980), "Machine-component group formation in group technology", *OMEGA: The International Journal of Management Science*, **8**, 193-199.

King, J.R., and Nakornchai, V. (1982), "Machine-component group formation in group technology: review and extension", *International Journal of Production Research*, **20**, pp.117-133.

King, N., and Majchrzak, A. (1996), "Concurrent Engineering tools: Are the human issues being ignored", *IEEE Transactions on Engineering Management*, **43**, pp.189-201.

Kitaoka, M., Nakamura, R., Serizawa, S., and Usuki, J. (1999), "Multivariate analysis model for machine-part cell formation problem in group technology", *International Journal of Production Economics*, **60-61**, pp.433-438.

Klippel, E.M., de Alvarenga, A.G., and Gomes, F.J.N. (1999), "A two-phase procedure for cell formation in manufacturing systems", *Integrated Manufacturing Systems*, **10**, pp.367-375.

Konz, S., and Johnson, S. (2000), *Work Design Industrial Ergonomics*, *5th ed.*, Halcomb Hathaway Publisher, Arizona.

Kumar, K. R., Kusiak, A., and Vannelli, A. (1986), "Grouping of parts and components in flexible manufacturing systems", *European Journal of Operational Research*, **24**, pp.387-397.

Kumar, R.V., and Vanelli, A., "Strategic subcontracting for efficient disaggrated manufacturing", *International Journal of Production Research*, **25**, pp.1715-1728.

Kusiak, A. (1987), "The generalized group technology concept", *International Journal of Production Research*, **25**, pp.561-569.

Kusiak, A., and Cho, M. (1992), "Similarity coefficient algorithms for solving the group technology", *International Journal of Production Research*, **30**, pp.2633-2646.

Lee, K.S., Park, H.S., and Chun, Y.H. (1996), "The validity of the revised NIOSH weight limit in a Korean young male population: a psychophysical approach", *International Journal of Industrial Ergonomics*, **18**, pp.181-186.

Lee, Kyung-Mi, Yamakawa, T., and Lee, Keon-Myung (1997), "Machine-part grouping for cellular manufacturing systems: a neural network approach", *1997 First International Conference on Knowledge-Based Intelligent Electronic Systems*, 21-23 May, Adelaide, Australia, pp.575-580.

Liang, M., and Zolfaghari, S. (1999), "Machine cell formation considering processing times and machine capacities: an ortho-synapse Hopfield neural network approach", *Journal of Intelligence Manufacturing*, **10**, pp.437-447.

Liao, T.W., and Gen, L.J. (1993), "An evaluation of ART1 neural network models for GT part family and machine cell forming", *Journal of Manufacturing Systems*, **12**, pp.282.

Liao, T.W., and Lee, K.S. (1994), "Integration of a feature-based CAD system and an ART1 neural model for GT coding and part family forming", *Computers and Industrial Engineering*, **26**, 93.

Logendran, R. (1990), "A workload based model for minimizing total intercell and intracell moves in cellular manufacturing", *International Journal of Production Research*, **28**, pp.913-925.

Logendran, R. (1991), "Impact of sequence of operations and layout of cells in cellular manufacturing", *International Journal of Production Research*, **29**, pp.375-390.

Logendran R., and West, T.M. (1990), "A machine-part based grouping algorithm in cellular manufacturing", *Computers and Industrial Engineering*, **19**, pp.57-61.

Maffei, M.J., and Meredith, J. (1994), "The organizational side of flexible manufacturing technology: guidelines for managers", *International Journal of Operations & Production Management*, **14**, pp.17-34.

Mahdavi, I., Kaushal, O.P., and Chandra, M. (2001), "Graph-neural network approach in cellular manufacturing on the basis of a binary system", *International Journal of Production Research*, **39**, pp.2913-2922.

Mak, K.L., and Wong, Y.S. (2000), "Genetic design of cellular manufacturing systems", *Human Factors and Ergonomics in Manufacturing*, **10**, pp.177-192.

McAuley, J. (1972), "Machine grouping for efficient production", *The Production Engineer*, **51**, pp.53.

McCormick, Jr., W.T., Schweitzer, P.J., and White, T.W. (1972), "Problem decomposition and data reorganization by a clustering technique", *Operations Research*, **20**, pp.993-1009.

Min, H., and Shin, D. (1993), "Simultaneous formation of machine and human cells in group technology: a multiple objective approach", *International Journal of Production Research*, **31**, pp.2307-2318.

Molleman, E., Slomp, J., and Rolefes, S. (2002), "The evolution of a cellular manufacturing systems – a longitudinal study", *International Journal of Production Economics*, **75**, pp.305-322.

Moon, C., and Gen, M. (1999), "A genetic algorithm-based approach for design of independent manufacturing cells", *International Journal of Production Economics*, **60-61**, pp.421-426.

Mukattash, A.M., Adil, M.B., and Tahboub, K.K. (2002), "Heuristic approaches for part assignment in cell formation", *Computers and Industrial Engineering*, **42**, pp.329-341.

Nagi, R., Harhalakis, G., and Proth, J. M. (1990), "Multiple routeings and capacity considerations in group technology applications", *International Journal of Production Research*, **28**, pp.2243-2257.

Norman, B.A., Tharmmaphornphilas, W., Needy, K.L., Bidanda, B., and Warner, R.C. (2002), "Worker assignment in cellular manufacturing considering technical and human skills", *International Journal of Production Research*, **40**, pp.1479-1492.

Okogbaa, O. G., Chen, M. T., Changchit, C., and Shell, R. L. (1992), "Manufacturing system cell formation and evaluation using a new inter-cell from reduction heuristic", *International Journal of Production Research*, **30**, pp.1101-1118.

Olorunniwo, F., and Udo, G. (2002), "The impact of management and employees on cellular manufacturing implementation", *International Journal of Production Economics*, **76**, pp.27-38.

Onwubolu, G.C., and Mutingi, M. (2001), "A genetic algorithm approach to cellular manufacturing systems", *Computers and Industrial Engineering*, **39**, pp.125-144.

Ozdemir, A.A. (1995), *Simultaneous part family and machine cell formations in cellular manufacturing systems: an analytical and algorithmic approach*, Master thesis, Department of Industrial and Manufacturing Systems, University of Windsor, Windsor, Ontario.

Pilot, T., and Knosala, R. (1998), "The application of neural networks in group technology", *Journal of Materials Processing Technology*, **78**, pp.150-155.

Pullen, R.D. (1976), "A survey of cellular manufacturing cells", *The Production Engineer*, **55**, pp.451.

Purcheck, G.F.K. (1975), "A mathematical classification as a basis for the design of group-technology production cells", *The Production Engineer*, **51**, pp.35.

Purcheck, G. (1985), "Machine-component group formation: an heuristic method for flexible production cells and flexible manufacturing systems", *International Journal of Production Research*, **23**, pp.911-943.

Rajamani, D., Singh, N., and Aneja, Y. P. (1990), "Integrated design of cellular manufacturing systems in the presence of alternative process plans", *International Journal of Production Research*, **28**, pp.1541-1554.

Rajagopalan, R., and Batra, J. (1975), "Design of cellular production systems – a graph theoretic approach", *International Journal of Production Research*, **13**, pp.567-579.

Ramabhatta, V., and Nagi, R. (1998), "An integrated formulation of manufacturing cell formation with capacity planning and multiple routings", *Annals of Operations Research*, **77**, pp.79-95.

Salum, L. (2000), "The cellular manufacturing layout problem", *International Journal of Production Research*, **38**, pp.1053-1069.

Sarker, B.R., Li, K. (1997), "Simultaneous route selection and cell formation: a mixed-integer programming time-cost model", *Integrated Manufacturing Systems*, **8**, pp.374-377.

Seifoddini, H., and Wolfe, P.M. (1986), "Application of the similarity coefficient method in group technology", *IIE Transactions*, **18**, pp.271-277.

Seifoddini, H., and Hsu, C. (1994), "Comparative study of similarity coefficients and clustering algorithms in cellular manufacturing", *Journal of Manufacturing Systems*, **13**, pp.119-127.

Selim, H.M., Askin, R.G., and Vakharia, A.J. (1998), "Cell formation in group technology: review, evaluation and directions for future research", *Computers and Industrial Engineering*, **34**, pp.3-20.

Shafer, S. M., Kern, D. M., and Wei, J. C. (1992), "A mathematical programming approach for dealing with exceptional elements in cellular manufacturing", *International Journal of Production Research*, **30**, pp.1029-1036.

Shafer, S.M., and Rogers, D.F. (1993), "Similarity and distance measures for cellular manufacturing. Part I. A survey, *International Journal of Production Research*, **31**, pp.1133-1142.

Shafer, S.M., and Rogers, D.F. (1993), "Similarity and distance measures for cellular manufacturing. Part II. An extension and comparison, *International Journal of Production Research*, **31**, pp.1315-1326.

Shiko, G. (1992), "A process planning-oriented approach to part family formation problem in group technology applications", *International Journal of Production Research*, **30**, pp.1739-1752.

Singh, N., and Mohanty, B. K. (1991), "A fuzzy approach to multi-objective routing problem with applications to process planning in manufacturing", *International Journal of Production Research*, **29**, pp.1161-1170.

Sobhanallahi, M.A., Jahanshahloo, G.R., Amin, G.R., and Shayan E. (2002), "Threshold value for the number of cells in group technology", *Computers and Industrial Engineering*, **42**, pp.231-236.

Sofianopoulou, S. (1999), "Manufacturing cells design with alternative process plans and/or replicated machines", *International Journal of Production Research*, **37**, pp.707-720.

Sohal, A.S., Fitzpatrick, P., and Power, D. (2001), "A longitudinal study of a flexible manufacturing cell operation", *Integrated Manufacturing Systems*, **12**, pp.236-245.

Song, S., and Hitomi, K. (1992), "GT cell formation for minimizing the intercell parts flow", *International Journal of Production Research*, **30**, pp.2737-2753.

Süer, G.A. (1996), "Optimal operator assignment and cell loading in labor intensive manufacturing cells", *Computers and Industrial Engineering*, **31**, pp.155-158.

Süer, G.A., and Ortega, M. (1994), "A machine level based similarity coefficient for forming manufacturing cells", *Computers and Industrial Engineering*, **27**, pp.67-70.

Taboun, S.M., Sankaran, S., and Bhole, S. (1991), "Comparison and evaluation of similarity measures in group technology, *Computers and Industrial Engineering*, **20**, pp.343-353.

Tatikonda, M. V., and Wemmerlöv, U. (1992), "Adoption and implementation of group technology classification and coding systems: insights from seven case studies", *International Journal of Production Research*, **30**, pp.2087-2110.

Tsai, C.C., Chu, C.H., and Barta, T.A. (1997), "Modeling and analysis of a manufacturing cell formation problem with fuzzy mixed-integer programming", *IIE Transactions*, **29**, pp.533-547.

Vakharia, A.J., and Chang, Y.L. (1997), "Cell formation in group technology: a combinatorial search approach", *International Journal of Production Research*, **35**, pp.2025-2043.

Vohra, T., Chen, D. S., Chang, J. C., and Chen, H. C. (1990), "A network approach to cell formation in cellular manufacturing", *International Journal of Production Research*, **28**, pp.2075-2084.

Waghodekar, P.H., and Sahu, S. (1984), "Machine-component cell formation in group technology: MACE', *International Journal of Production Research*, **22**, pp.937-948.

Wang, M.J., Garg, A., Chang, Y.C., Shih, Y.C., Yeh, W.Y., and Lee, C.L. (1998), "The relationship between low back discomfort ratings and the NIOSH lifting index", *Human Factors*, **40**, pp.509-515.

Warner, R.C., Needy, K.L., and Bidanda B. (1997), "Worker assignment in implementing manufacturing cells", *Proceedings of 6th Industrial Engineering Research Conference*, pp.240-245.

Waters, T.R., and Putz-Anderson, V. (1998), "Assessment of manual lifting – the NIOSH approach", *Ergonomics in Manufacturing: Raising Productivity through Workplace Improvement*, ed. W. Karwowski and G. Salvendy, Society of Manufacturing Engineers, Dearborn, Michigan, pp.204-241.

Waters, T.R., Baron, S.L., and Kemmlert, K. (1998), "Accuracy of measurements for the revised NIOSH lifting equation", *Applied Ergonomics*, **29**, pp.433-438.

Wei, J.C., and Kern, G.M. (1989), "Commonality analysis: a linear cell clustering algorithm for group technology", *International Journal of Production Research*, **27**, pp.2053-2062.

Wemmerlöv, U., and Hyer, N.L. (1989), "Cellular manufacturing in the U.S. industry: a survey of users", *International Journal of Production Research*, **27**, pp.1511-1530.

Wemmerlöv, U., and Johnson, D.J. (1997), "Cellular manufacturing at 46 user plants: implementation experiences and performance improvements, *International Journal of Production Research*, **35**, pp.29-49.

Won, Y. (2000), "New *p*-median approach to cell formation with alternative process plans", *International Journal of Production Research*, **38**, pp.229-240.

Won, Y., and Kim, S. (1997), "Multiple criteria clustering algorithm for solving the group technology problems with multiple process routings", *Computers and Industrial Engineering*, **32**, pp.207-220.

Won, Y., and Lee, K.C. (2001), "Group technology cell formation considering operation sequences and production volumes", *International Journal of Production Research*, **39**, pp.2755-2768.

Wu, N., and Salvendy, G. (1993), "A modified network approach for the design of cellular manufacturing systems", *International Journal of Production Research*, **31**, pp.1409-1421.

Zhou, M., and Askin, R.G. (1998), "Formation of general GT cells: an operation-based approach", *Computers and Industrial Engineering*, **34**, pp.147-157.

Appendix I

LINGO Source Codes

A.1.1. LINGO Source Code: Example 1

```
! MODEL 1 - EXAMPLE 1 ;

MODEL:
SETS:
        PART    /1..8/: D, LW, LD;
        MACHINE /1..5/: F, JI, G;
        CELL    /1..2/: S, VW, W, FREQ, CLI, WI,INCR, ASSIGN;
        PM (PART,MACHINE): T,L;
        PC (PART,CELL)   : X, FM, STLI;
        MC (MACHINE,CELL): MA, MI;

ENDSETS

DATA:
!Import Example Data from Excel;
D, LW, LD, F, G, JI, S, VW, T = @OLE('D:\EX1THESIS.XLS', 'Demand', 'Load_Weight',
'Lifting_Distance', 'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost',
'Cell_Capacity', 'Max_Worker', 'Processing_Time');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 15000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX1THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + LIFTING_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MC(M,C): F(M)*MA(M,C));
MACHINE_IDLE_COST = @SUM(MC(M,C): JI(M)*MI(M,C)/R);

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CELL(C): N*U*W(C));
WORKER_IDLE_COST = @SUM(CELL(C): JW*WI(C)/R);

! Lifting penalty cost;
LIFTING_COST = LC * @SUM(CELL(C): W(C) * CLI(C)) ;

! Constraint A: 1 job to 1 cell;
@FOR(PART(P): @SUM(CELL(C): X(P,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MC(M,C): MI(M,C) + @SUM(PART(P): D(P)*T(P,M)*X(P,C)) = MA(M,C)*U*R );

! Constraint C: Cell capacity;
@FOR(CELL(C): @SUM(MACHINE(M): MA(M,C)) <= S(C));
@FOR(CELL(C): W(C) <= VW(C));

! Constraint E: Enough worker capacity;
@FOR(CELL(C): WI(C) + @SUM(PM(P,M): G(M)*D(P)*T(P,M)*X(P,C)) = W(C)*U*R );

! Constraint F: Lifting Frequency Range;
```

```
@FOR(CELL(C): FREQ(C) = @SUM(PART(P): X(P,C)*D(P)*@SUM(MACHINE(M): L(P,M))/U/R/W(C)) );
@FOR(CELL(C): FREQ(C) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PM(P,M)|T(P,M)#NE#0 : L(P,M) = 1 );
@FOR(PM(P,M)|T(P,M)#EQ#0 : L(P,M) = 0 );
@FOR(PC(P,C) : FM(P,C) = 0.8359-(0.0893464*X(P,C)*D(P)*@SUM(MACHINE(M): L(P,M))/U/R/W(C))
);
@FOR(PC(P,C) : STLI(P,C) = X(P,C)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,C)) );
@FOR(CELL(C): INCR(C) = 0.25*(@SUM(PART(P):STLI(P,C))- @MAX(PART(P): STLI(P,C))) /
(@SUM(PART(P): X(P,C))-1));
@FOR(CELL(C): CLI(C) = @MAX(PART(P): STLI(P,C)) + INCR(C));
@FOR(CELL(C): CLI(C) <= 1.5);

! Constraint H: 0/1 Integer;
@FOR(PC(P,C): @BIN(X(P,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MC(M,C): MA(M,C) >= 0 );
@FOR(MC(M,C): MI(M,C) >= 0 );
@FOR(CELL(C): W(C) >= 0 );
@FOR(CELL(C): WI(C) >= 0 );

! Constraint J: General integer;
@FOR(MC(M,C): @GIN(MA(M,C)) );
@FOR(CELL(C): @GIN(W(C)) );

END
```

A.1.2. LINGO Source Code: Example 2

```
! MODEL 1 - EXAMPLE 2 ;

MODEL:
SETS:
        PART    /1..15/: D, LW, LD;
        MACHINE /1..10/: F, JI, G;
        CELL    /1..3/ : S, VW, W, FREQ, CLI, WI,INCR;
        PM (PART,MACHINE): T,L;
        PC (PART,CELL)   : X, FM, STLI;
        MC (MACHINE,CELL): MA, MI;

ENDSETS

DATA:
!Import Example Data from Excel;
D, LW, LD, F, G, JI, S, VW, T = @OLE('D:\EX2THESIS.XLS', 'Demand', 'Load_Weight',
'Lifting_Distance', 'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost',
'Cell_Capacity', 'Max_Worker', 'Processing_Time');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 15000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
```

```
@OLE('D:\EX2THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + LIFTING_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MC(M,C): F(M)*MA(M,C));
MACHINE_IDLE_COST = @SUM(MC(M,C): JI(M)*MI(M,C)/R);

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CELL(C): N*U*W(C));
WORKER_IDLE_COST = @SUM(CELL(C): JW*WI(C)/R);

! Lifting penalty cost;
LIFTING_COST = LC * @SUM(CELL(C): W(C) * CLI(C)) ;

! Constraint A: 1 job to 1 cell;
@FOR(PART(P): @SUM(CELL(C): X(P,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MC(M,C): MI(M,C) + @SUM(PART(P): D(P)*T(P,M)*X(P,C)) = MA(M,C)*U*R );

! Constraint C: Cell capacity;
@FOR(CELL(C): @SUM(MACHINE(M): MA(M,C)) <= S(C));
@FOR(CELL(C): W(C) <= VW(C));

! Constraint E: Enough worker capacity;
@FOR(CELL(C): WI(C) + @SUM(PM(P,M): G(M)*D(P)*T(P,M)*X(P,C)) = W(C)*U*R );

! Constraint F: Lifting Frequency Range;
@FOR(CELL(C): FREQ(C) = @SUM(PART(P): X(P,C)*D(P)*@SUM(MACHINE(M): L(P,M))/U/R/W(C)) );
@FOR(CELL(C): FREQ(C) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PM(P,M)|T(P,M)#NE#0 : L(P,M) = 1 );
@FOR(PM(P,M)|T(P,M)#EQ#0 : L(P,M) = 0 );
@FOR(PC(P,C) : FM(P,C) = 0.8359-(0.0893464*X(P,C)*D(P)*@SUM(MACHINE(M): L(P,M))/U/R/W(C))
);
@FOR(PC(P,C) : STLI(P,C) = X(P,C)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,C)) );
@FOR(CELL(C): INCR(C) = 0.25*(@SUM(PART(P):STLI(P,C))- @MAX(PART(P): STLI(P,C))) /
(@SUM(PART(P): X(P,C))-1));
@FOR(CELL(C): CLI(C) = @MAX(PART(P): STLI(P,C)) + INCR(C));
@FOR(CELL(C): CLI(C) <= 1.5);

! Constraint H: 0/1 Integer;
@FOR(PC(P,C): @BIN(X(P,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MC(M,C): MA(M,C) >= 0 );
@FOR(MC(M,C): MI(M,C) >= 0 );
@FOR(CELL(C): W(C) >= 0 );
@FOR(CELL(C): WI(C) >= 0 );

! Constraint J: General integer;
@FOR(MC(M,C): @GIN(MA(M,C)) );
@FOR(CELL(C): @GIN(W(C)) );

END
```

A.1.3. LINGO Source Code: Example 3

```
! MODEL 2 - EXAMPLE 3 ;

MODEL:
SETS:
        JOB     /1..4/: ;
        PART    /1..9/: D, LW, LD, H;
        MACHINE /1..6/: F, JI, G;
        CELL    /1..2/: S, VW, W, FREQ, CLI, WI,INCR;
        PJM (PART,JOB,MACHINE): T;
        PJC (PART,JOB,CELL)   : X, STLI, V, VN,FM;
        MC (MACHINE,CELL)     : MA, MI;
        PJ (PART,JOB)         : L;

ENDSETS

DATA:
!Import Example Data from Excel;
D, LW, LD, F, G, JI, S, VW, T, H = @OLE('D:\EX3THESIS.XLS', 'Demand', 'Load_Weight',
'Lifting_Distance', 'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost',
'Cell_Capacity', 'Max_Worker', 'Processing_Time', 'Intercell_Move_Cost');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 15000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX3THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA


! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + INTERCELL_COST + WORKER_COST + WORKER_IDLE_COST
+ LIFTING_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MC(M,C): F(M)*MA(M,C));
MACHINE_IDLE_COST = @SUM(MC(M,C): JI(M)*MI(M,C)/R);

! Intercell Movements Cost;
INTERCELL_COST = @SUM(PJC(P,J,C)|J#GE#2: H(P)*D(P)*V(P,J,C));

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CELL(C): N*U*W(C));
WORKER_IDLE_COST = @SUM(CELL(C): JW*WI(C)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CELL(C): LC * W(C) * CLI(C)) ;

! Constraint A: 1 job to 1 cell;
@FOR(PJ(P,J): @SUM(CELL(C): X(P,J,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MC(M,C): MI(M,C) + @SUM(PJ(P,J): D(P)*T(P,J,M)*X(P,J,C)) = MA(M,C)*U*R );

! Constraint C: Cell capacity;
```

```
@FOR(CELL(C): @SUM(MACHINE(M): MA(M,C)) <= S(C));
@FOR(CELL(C): W(C) <= VW(C));

! Constraint D: Intercell movements;
@FOR(PJC(P,J,C)|J#GE#2: X(P,J,C) - X(P,J-1,C) = V(P,J,C) - VN(P,J,C));

! Constraint E: Enough worker capacity;
@FOR(CELL(C): WI(C) + @SUM(PJM(P,J,M): G(M)*D(P)*T(P,J,M)*X(P,J,C)) = W(C)*U*R );

! Constraint F: Lifting Frequency Range;
@FOR(PJ(P,J)|@SUM(MACHINE(M):T(P,J,M))#NE#0 : L(P,J) = 1 );
@FOR(PJ(P,J)|@SUM(MACHINE(M):T(P,J,M))#EQ#0 : L(P,J) = 0 );
@FOR(CELL(C): FREQ(C) = @SUM(PJ(P,J): D(P)*X(P,J,C)*L(P,J)/U/R/W(C)) );
@FOR(CELL(C): FREQ(C) <= 3 );

! Constraint G: Composite Lifting Index;
@FOR(PJC(P,J,C) : FM(P,J,C) = 0.8359-(0.0893464*D(P)*X(P,J,C)*L(P,J)/U/R/W(C)) );
@FOR(PJC(P,J,C) : STLI(P,J,C) = X(P,J,C)*L(P,J)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,J,C))
);
@FOR(CELL(C): INCR(C) = 0.25*(@SUM(PJ(P,J):STLI(P,J,C))- @MAX(PJ(P,J): STLI(P,J,C))) /
(@SUM(PJ(P,J): X(P,J,C)*L(P,J))-1));
@FOR(CELL(C): CLI(C) = @MAX(PJ(P,J): STLI(P,J,C)) + INCR(C));
@FOR(CELL(C): CLI(C) <= 1.5);

! Constraint H: 0/1 Integer;
@FOR(PJC(P,J,C): @BIN(X(P,J,C)) );
@FOR(PJC(P,J,C): @BIN(V(P,J,C)) );
@FOR(PJC(P,J,C): @BIN(VN(P,J,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MC(M,C): MA(M,C) >= 0 );
@FOR(MC(M,C): MI(M,C) >= 0 );
@FOR(CELL(C): W(C) >= 1 );
@FOR(CELL(C): WI(C) >= 0 );

! Constraint J: General integer;
@FOR(MC(M,C): @GIN(MA(M,C)) );
@FOR(CELL(C): @GIN(W(C)) );

END
```

A.1.4. LINGO Source Code: Example 4

```
! MODEL 2 - EXAMPLE 4 ;

MODEL:
SETS:
       JOB     /1..4/ : ;
       PART    /1..20/: D, LW, LD, H;
       MACHINE /1..12/: F, JI, G;
       CELL    /1..3/ : S, VW, W, FREQ, CLI, WI,INCR;
       PJM (PART,JOB,MACHINE): T;
       PJC (PART,JOB,CELL)   : X, STLI, V, VN,FM;
       MC (MACHINE,CELL)     : MA, MI;
       PJ (PART,JOB)         : L;
ENDSETS

DATA:
!Import Example Data from Excel;
D, LW, LD, F, G, JI, S, VW, T, H = @OLE('D:\EX4THESIS.XLS', 'Demand', 'Load_Weight',
'Lifting_Distance', 'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost',
'Cell_Capacity', 'Max_Worker', 'Processing_Time', 'Intercell_Move_Cost');

! Worker Salary / hour;
N = 10;
```

```
! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 15000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX4THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA


! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + INTERCELL_COST + WORKER_COST + WORKER_IDLE_COST
+ LIFTING_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MC(M,C): F(M)*MA(M,C));
MACHINE_IDLE_COST = @SUM(MC(M,C): JI(M)*MI(M,C)/R);

! Intercell Movements Cost;
INTERCELL_COST = @SUM(PJC(P,J,C)|J#GE#2: H(P)*D(P)*V(P,J,C));

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CELL(C): N*U*W(C));
WORKER_IDLE_COST = @SUM(CELL(C): JW*WI(C)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CELL(C): LC * W(C) * CLI(C)) ;

! Constraint A: 1 job to 1 cell;
@FOR(PJ(P,J): @SUM(CELL(C): X(P,J,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MC(M,C): MI(M,C) + @SUM(PJ(P,J): D(P)*T(P,J,M)*X(P,J,C)) = MA(M,C)*U*R );

! Constraint C: Cell capacity;
@FOR(CELL(C): @SUM(MACHINE(M): MA(M,C)) <= S(C));
@FOR(CELL(C): W(C) <= VW(C));

! Constraint D: Intercell movements;
@FOR(PJC(P,J,C)|J#GE#2: X(P,J,C) - X(P,J-1,C) = V(P,J,C) - VN(P,J,C));

! Constraint E: Enough worker capacity;
@FOR(CELL(C): WI(C) + @SUM(PJM(P,J,M): G(M)*D(P)*T(P,J,M)*X(P,J,C)) = W(C)*U*R );

! Constraint F: Lifting Frequency Range;
@FOR(PJ(P,J)|@SUM(MACHINE(M):T(P,J,M))#NE#0 : L(P,J) = 1 );
@FOR(PJ(P,J)|@SUM(MACHINE(M):T(P,J,M))#EQ#0 : L(P,J) = 0 );
@FOR(CELL(C): FREQ(C) = @SUM(PJ(P,J): D(P)*X(P,J,C)*L(P,J)/U/R/W(C)) );
@FOR(CELL(C): FREQ(C) <= 3 );

! Constraint G: Composite Lifting Index;
@FOR(PJC(P,J,C) : FM(P,J,C) = 0.8359-(0.0893464*D(P)*X(P,J,C)*L(P,J)/U/R/W(C)) );
@FOR(PJC(P,J,C) : STLI(P,J,C) = X(P,J,C)*L(P,J)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,J,C))
);
@FOR(CELL(C): INCR(C) = 0.25*(@SUM(PJ(P,J):STLI(P,J,C))- @MAX(PJ(P,J): STLI(P,J,C))) /
(@SUM(PJ(P,J): X(P,J,C)*L(P,J))-1));
@FOR(CELL(C): CLI(C) = @MAX(PJ(P,J): STLI(P,J,C)) + INCR(C));
@FOR(CELL(C): CLI(C) <= 1.5);

! Constraint H: 0/1 Integer;
@FOR(PJC(P,J,C): @BIN(X(P,J,C)) );
```

```
@FOR(PJC(P,J,C): @BIN(V(P,J,C)) );
@FOR(PJC(P,J,C): @BIN(VN(P,J,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MC(M,C): MA(M,C) >= 0 );
@FOR(MC(M,C): MI(M,C) >= 0 );
@FOR(CELL(C): W(C) >= 1 );
@FOR(CELL(C): WI(C) >= 0 );

! Constraint J: General integer;
@FOR(MC(M,C): @GIN(MA(M,C)) );
@FOR(CELL(C): @GIN(W(C)) );

END
```

A.1.5. LINGO Source Code: Example 5

```
! MODEL 3 - EXAMPLE 5 ;

MODEL:
SETS:
        PART    /1..10/: H,B,LW, LD;
        MACHINE /1..7/ : F, G, JI;
        CELL    /1..2/ : ;
        PERIOD  /1..3/ : IM, DM, IW, DW;
        PT   (PART,PERIOD)        : D;
        PM   (PART,MACHINE)       : OT,L;
        PTC (PART,PERIOD,CELL)    : X, FILI, STLI, FM,RWL;
        MT   (MACHINE,PERIOD)     : ;
        MTC (MACHINE,PERIOD,CELL) : MA,MI,DPM,DNM;
        CT   (CELL,PERIOD)              : S, V, W, WI, DPW, DNW, FREQ, CLI, INCR;
ENDSETS

DATA:
!Import Example Data from Excel;
D, F, G, JI, S, V, OT, IM, DM, IW, DW, LW, LD = @OLE('D:\EX5THESIS.XLS', 'Demand',
'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost', 'Cell_Capacity', 'Max_Worker',
'Processing_Time', 'IM', 'DM', 'IW', 'DW', 'Load_Weight', 'Lifting_Distance');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 20000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX5THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + LIFTING_COST +
MCHANGE_COST + WCHANGE_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MTC(M,T,C): F(M)*MA(M,T,C));
MACHINE_IDLE_COST = @SUM(MTC(M,T,C): JI(M)*MI(M,T,C)/R);
```

```
! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CT(C,T): N*U*W(C,T));
WORKER_IDLE_COST = @SUM(CT(C,T): JW*WI(C,T)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CT(C,T): LC * W(C,T) * CLI(C,T)) ;

! Machine Capacity Change;
MCHANGE_COST = @SUM(MTC(M,T,C): IM(T)*DPM(M,T,C) + DM(T)*DNM(M,T,C) );

! Worker Capacity Change;
WCHANGE_COST = @SUM(CT(C,T): IW(T)*DPW(C,T) + DW(T)*DNW(C,T) );

! Constraint A: 1 job to 1 cell;
@FOR(PT(P,T): @SUM(CELL(C): X(P,T,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MTC(M,T,C): MI(M,T,C) + @SUM(PART(P): D(P,T)*OT(P,M)*X(P,T,C)) = MA(M,T,C)*U*R );

! Constraint C: Cell capacity;
@FOR(CT(C,T): @SUM(MACHINE(M): MA(M,T,C)) <= S(C,T));
@FOR(CT(C,T): W(C,T) <= V(C,T));

! Constraint E: Enough worker capacity;
@FOR(CT(C,T): WI(C,T) + @SUM(PM(P,M): G(M)*D(P,T)*OT(P,M)*X(P,T,C)) = W(C,T)*U*R );

! Constraint F: Lifting Frequency Range;
@FOR(PM(P,M)|OT(P,M)#NE#0 : L(P,M) = 1 );
@FOR(PM(P,M)|OT(P,M)#EQ#0 : L(P,M) = 0 );
@FOR(CT(C,T): FREQ(C,T) = @SUM(PART(P): X(P,T,C)*D(P,T)*@SUM(MACHINE(M):
L(P,M))/U/R/W(C,T)) );
@FOR(CT(C,T): FREQ(C,T) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PTC(P,T,C) : FM(P,T,C) = 0.8359-(0.0893464*X(P,T,C)*D(P,T)*@SUM(MACHINE(M):
L(P,M))/U/R/W(C,T)) );
@FOR(PTC(P,T,C) : STLI(P,T,C) = X(P,T,C)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,T,C)) );
@FOR(CT(C,T): INCR(C,T) = 0.25*(@SUM(PART(P):STLI(P,T,C))- @MAX(PART(P): STLI(P,T,C))) /
(@SUM(PART(P): X(P,T,C))-1));
@FOR(CT(C,T): CLI(C,T) = @MAX(PART(P): STLI(P,T,C)) + INCR(C,T));
@FOR(CT(C,T): CLI(C,T) <= 1.5);


! Constraint G: Machine & Worker Balance Equations;
@FOR(MTC(M,T,C)|T#GE#2: MA(M,T,C) - MA(M,T-1,C) = DPM(M,T,C) - DNM(M,T,C));
@FOR(CT(C,T)|T#GE#2: W(C,T) - W(C,T-1) = DPW(C,T) - DNW(C,T));

! Constraint H: 0/1 Integer;
@FOR(PTC(P,T,C): @BIN(X(P,T,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MTC(M,T,C): MA(M,T,C) >= 0 );
@FOR(MTC(M,T,C): MI(M,T,C) >= 0 );
@FOR(CT(C,T): W(C,T) >= 0 );
@FOR(CT(C,T): WI(C,T) >= 0 );
@FOR(MTC(M,T,C): DPM(M,T,C) >= 0);
@FOR(MTC(M,T,C): DNM(M,T,C) >= 0);
@FOR(CT(C,T): DPW(C,T) >= 0 );
@FOR(CT(C,T): DNW(C,T) >= 0 );

! Constraint J: General integer;
@FOR(MTC(M,T,C): @GIN(MA(M,T,C)) );
@FOR(CT(C,T): @GIN(W(C,T)) );
@FOR(MTC(M,T,C): @GIN(DPM(M,T,C)) );
@FOR(MTC(M,T,C): @GIN(DNM(M,T,C)) );
@FOR(CT(C,T): @GIN(DPW(C,T)) );
@FOR(CT(C,T): @GIN(DNW(C,T)) );

END
```

A.1.6. LINGO Source Code: Example 6

```
! MODEL 3 - EXAMPLE 6 ;

MODEL:
SETS:
        PART    /1..20/: H,B,LW, LD;
        MACHINE /1..12/: F, G, JI;
        CELL    /1..3/ : ;
        PERIOD  /1..4/ : IM, DM, IW, DW;
        PT  (PART,PERIOD)        : D;
        PM  (PART,MACHINE)       : OT,L;
        PTC (PART,PERIOD,CELL)     : X, FILI, STLI, FM,RWL;
        MT  (MACHINE,PERIOD)     : ;
        MTC (MACHINE,PERIOD,CELL) : MA,MI,DPM,DNM;
        CT  (CELL,PERIOD)               : S, W, WI, DPW, DNW, FREQ, CLI, INCR;
ENDSETS

DATA:
!Import Example Data from Excel;
D, F, G, JI, S, V, OT, IM, DM, IW, DW, LW, LD = @OLE('D:\EX6THESIS.XLS', 'Demand',
'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost', 'Cell_Capacity', 'Max_Worker',
'Processing_Time', 'IM', 'DM', 'IW', 'DW', 'Load_Weight', 'Lifting_Distance');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 20000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX6THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + LIFTING_COST +
MCHANGE_COST + WCHANGE_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MTC(M,T,C): F(M)*MA(M,T,C));
MACHINE_IDLE_COST = @SUM(MTC(M,T,C): JI(M)*MI(M,T,C)/R);

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CT(C,T): N*U*W(C,T));
WORKER_IDLE_COST = @SUM(CT(C,T): JW*WI(C,T)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CT(C,T): LC * W(C,T) * CLI(C,T)) ;

! Machine Capacity Change;
MCHANGE_COST = @SUM(MTC(M,T,C): IM(T)*DPM(M,T,C) + DM(T)*DNM(M,T,C) );

! Worker Capacity Change;
WCHANGE_COST = @SUM(CT(C,T): IW(T)*DPW(C,T) + DW(T)*DNW(C,T) );

! Constraint A: 1 job to 1 cell;
@FOR(PT(P,T): @SUM(CELL(C): X(P,T,C)) = 1 );
```

```
! Constraint B: Enough machining time;
@FOR(MTC(M,T,C): MI(M,T,C) + @SUM(PART(P): D(P,T)*OT(P,M)*X(P,T,C)) = MA(M,T,C)*U*R );

! Constraint C: Cell capacity;
@FOR(CT(C,T): @SUM(MACHINE(M): MA(M,T,C)) <= S(C,T));
@FOR(CT(C,T): W(C,T) <= V(C,T));

! Constraint E: Enough worker capacity;
@FOR(CT(C,T): WI(C,T) + @SUM(PM(P,M): G(M)*D(P,T)*OT(P,M)*X(P,T,C)) = W(C,T)*U*R );

! Constraint F: Lifting Frequency Range;
@FOR(PM(P,M)|OT(P,M)#NE#0 : L(P,M) = 1 );
@FOR(PM(P,M)|OT(P,M)#EQ#0 : L(P,M) = 0 );
@FOR(CT(C,T): FREQ(C,T) = @SUM(PART(P): X(P,T,C)*D(P,T)*@SUM(MACHINE(M):
L(P,M))/U/R/W(C,T)) );
@FOR(CT(C,T): FREQ(C,T) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PTC(P,T,C) : FM(P,T,C) = 0.8359-(0.0893464*X(P,T,C)*D(P,T)*@SUM(MACHINE(M):
L(P,M))/U/R/W(C,T)) );
@FOR(PTC(P,T,C) : STLI(P,T,C) = X(P,T,C)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,T,C)) );
@FOR(CT(C,T): INCR(C,T) = 0.25*(@SUM(PART(P):STLI(P,T,C))- @MAX(PART(P): STLI(P,T,C))) /
(@SUM(PART(P): X(P,T,C))-1));
@FOR(CT(C,T): CLI(C,T) = @MAX(PART(P): STLI(P,T,C)) + INCR(C,T));
@FOR(CT(C,T): CLI(C,T) <= 1.5);


! Constraint G: Machine & Worker Balance Equations;
@FOR(MTC(M,T,C)|T#GE#2: MA(M,T,C) - MA(M,T-1,C) = DPM(M,T,C) - DNM(M,T,C));
@FOR(CT(C,T)|T#GE#2: W(C,T) - W(C,T-1) = DPW(C,T) - DNW(C,T));

! Constraint H: 0/1 Integer;
@FOR(PTC(P,T,C): @BIN(X(P,T,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MTC(M,T,C): MA(M,T,C) >= 0 );
@FOR(MTC(M,T,C): MI(M,T,C) >= 0 );
@FOR(CT(C,T): W(C,T) >= 0 );
@FOR(CT(C,T): WI(C,T) >= 0 );
@FOR(MTC(M,T,C): DPM(M,T,C) >= 0);
@FOR(MTC(M,T,C): DNM(M,T,C) >= 0);
@FOR(CT(C,T): DPW(C,T) >= 0 );
@FOR(CT(C,T): DNW(C,T) >= 0 );

! Constraint J: General integer;
@FOR(MTC(M,T,C): @GIN(MA(M,T,C)) );
@FOR(CT(C,T): @GIN(W(C,T)) );
@FOR(MTC(M,T,C): @GIN(DPM(M,T,C)) );
@FOR(MTC(M,T,C): @GIN(DNM(M,T,C)) );
@FOR(CT(C,T): @GIN(DPW(C,T)) );
@FOR(CT(C,T): @GIN(DNW(C,T)) );

END
```

A.1.7. LINGO Source Code: Example 7

```
! MODEL 4 - EXAMPLE 7 ;

MODEL:
SETS:
        PART    /1..10/: H,B,LW, LD;
        JOB      /1..3 /: ;
        MACHINE /1..7/ : F, G, JI;
        CELL    /1..2/ : ;
        PERIOD  /1..3/ : IM, DM, IW, DW;
```

```
        PT   (PART,PERIOD)                : D;
        PJT  (PART,JOB,PERIOD)            : ;
        PJM  (PART,JOB,MACHINE)           : OT;
        PJ   (PART,JOB)                   : L;
        PJTC(PART,JOB,PERIOD,CELL)    : X, FILI, STLI, FM,RWL,V,VN;
        MT   (MACHINE,PERIOD)             : ;
        MTC  (MACHINE,PERIOD,CELL)        : MA,MI,DPM,DNM;
        CT   (CELL,PERIOD)                      : S, VW, W, WI, DPW, DNW, FREQ, CLI, INCR;
ENDSETS

DATA:
!Import Example Data from Excel;
D, F, G, JI, S, VW, OT, IM, DM, IW, DW, LW, LD, H = @OLE('D:\EX7THESIS.XLS', 'Demand',
'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost', 'Cell_Capacity', 'Max_Worker',
'Processing_Time', 'IM', 'DM', 'IW', 'DW', 'Load_Weight', 'Lifting_Distance','IMC');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 20000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX7THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + INTERCELL_COST
+ LIFTING_COST + MCHANGE_COST + WCHANGE_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MTC(M,T,C): F(M)*MA(M,T,C));
MACHINE_IDLE_COST = @SUM(MTC(M,T,C): JI(M)*MI(M,T,C)/R);

! Intercell Movements Cost;
INTERCELL_COST = @SUM(PJTC(P,J,T,C)|J#GE#2: H(P)*D(P,T)*V(P,J,T,C));

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CT(C,T): N*U*W(C,T));
WORKER_IDLE_COST = @SUM(CT(C,T): JW*WI(C,T)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CT(C,T): LC * W(C,T) * CLI(C,T)) ;

! Machine Capacity Change;
MCHANGE_COST = @SUM(MTC(M,T,C): IM(T)*DPM(M,T,C) + DM(T)*DNM(M,T,C) );

! Worker Capacity Change;
WCHANGE_COST = @SUM(CT(C,T): IW(T)*DPW(C,T) + DW(T)*DNW(C,T) );

! Constraint A: 1 job to 1 cell;
@FOR(PJT(P,J,T): @SUM(CELL(C): X(P,J,T,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MTC(M,T,C): MI(M,T,C) + @SUM(PJ(P,J): D(P,T)*OT(P,J,M)*X(P,J,T,C)) = MA(M,T,C)*U*R
);

! Constraint C: Cell capacity;
@FOR(CT(C,T): @SUM(MACHINE(M): MA(M,T,C)) <= S(C,T));
@FOR(CT(C,T): W(C,T) <= VW(C,T));
```

```
! Constraint D: Intercell movements;
@FOR(PJTC(P,J,T,C)|J#GE#2: X(P,J,T,C) - X(P,J-1,T,C) = V(P,J,T,C) - VN(P,J,T,C));

! Constraint E: Enough worker capacity;
@FOR(CT(C,T): WI(C,T) + @SUM(PJM(P,J,M): G(M)*D(P,T)*OT(P,J,M)*X(P,J,T,C)) = W(C,T)*U*R
);

! Constraint F: Lifting Frequency Range;
@FOR(PJ(P,J)|@SUM(MACHINE(M):OT(P,J,M))#NE#0 : L(P,J) = 1 );
@FOR(PJ(P,J)|@SUM(MACHINE(M):OT(P,J,M))#EQ#0 : L(P,J) = 0 );
@FOR(CT(C,T): FREQ(C,T) = @SUM(PJ(P,J): D(P,T)*X(P,J,T,C)*L(P,J)/U/R/W(C,T)) );
@FOR(CT(C,T): FREQ(C,T) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PJTC(P,J,T,C) : FM(P,J,T,C) = 0.8359-(0.0893464*D(P,T)*X(P,J,T,C)*L(P,J)/U/R/W(C,T))
);
@FOR(PJTC(P,J,T,C) : STLI(P,J,T,C) =
X(P,J,T,C)*L(P,J)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,J,T,C)) );
@FOR(CT(C,T): INCR(C,T) = 0.25*(@SUM(PJ(P,J):STLI(P,J,T,C))- @MAX(PJ(P,J):
STLI(P,J,T,C))) / (@SUM(PJ(P,J): X(P,J,T,C)*L(P,J))-1));
@FOR(CT(C,T): CLI(C,T) = @MAX(PJ(P,J): STLI(P,J,T,C)) + INCR(C,T));
@FOR(CT(C,T): CLI(C,T) <= 1.5);

! Constraint G: Machine & Worker Balance Equations;
@FOR(MTC(M,T,C)|T#GE#2: MA(M,T,C) - MA(M,T-1,C) = DPM(M,T,C) - DNM(M,T,C));
@FOR(CT(C,T)|T#GE#2: W(C,T) - W(C,T-1) = DPW(C,T) - DNW(C,T));

! Constraint H: 0/1 Integer;
@FOR(PJTC(P,J,T,C): @BIN(X(P,J,T,C)) );
@FOR(PJTC(P,J,T,C): @BIN(V(P,J,T,C)) );
@FOR(PJTC(P,J,T,C): @BIN(VN(P,J,T,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MTC(M,T,C): MA(M,T,C) >= 0 );
@FOR(MTC(M,T,C): MI(M,T,C) >= 0 );
@FOR(CT(C,T): W(C,T) >= 0 );
@FOR(CT(C,T): WI(C,T) >= 0 );
@FOR(MTC(M,T,C): DPM(M,T,C) >= 0);
@FOR(MTC(M,T,C): DNM(M,T,C) >= 0);
@FOR(CT(C,T): DPW(C,T) >= 0 );
@FOR(CT(C,T): DNW(C,T) >= 0 );

! Constraint J: General integer;
@FOR(MTC(M,T,C): @GIN(MA(M,T,C)) );
@FOR(CT(C,T): @GIN(W(C,T)) );
@FOR(MTC(M,T,C): @GIN(DPM(M,T,C)) );
@FOR(MTC(M,T,C): @GIN(DNM(M,T,C)) );
@FOR(CT(C,T): @GIN(DPW(C,T)) );
@FOR(CT(C,T): @GIN(DNW(C,T)) );

END
```

A.1.8. LINGO Source Code: Example 8

```
! MODEL 4 - EXAMPLE 8 ;

MODEL:
SETS:
        PART    /1..20/: H,B,LW, LD;
        JOB      /1..4 /: ;
        MACHINE /1..12/: F, G, JI;
        CELL    /1..4/ : ;
        PERIOD  /1..3/ : IM, DM, IW, DW;
        PT   (PART,PERIOD)            : D;
```

```
        PJT   (PART,JOB,PERIOD)          : ;
        PJM   (PART,JOB,MACHINE)         : OT;
        PJ    (PART,JOB)                        : L;
        PJTC  (PART,JOB,PERIOD,CELL)     : X, FILI, STLI, FM,RWL,V,VN;
        MT    (MACHINE,PERIOD)           : ;
        MTC   (MACHINE,PERIOD,CELL)      : MA,MI,DPM,DNM;
        CT    (CELL,PERIOD)              : S, VW, W, WI, DPW, DNW, FREQ, CLI, INCR;
ENDSETS

DATA:
!Import Example Data from Excel;
D, F, G, JI, S, VW, OT, IM, DM, IW, DW, LW, LD, H = @OLE('D:\EX8THESIS.XLS', 'Demand',
'Procurement_Cost', 'Attention_Req.', 'Machine_Idle_Cost', 'Cell_Capacity', 'Max_Worker',
'Processing_Time', 'IM', 'DM', 'IW', 'DW', 'Load_Weight', 'Lifting_Distance','IMC');

! Worker Salary / hour;
N = 10;

! Total shop time / period, hours;
U = 2000;
R = 60;

! Lifting penalty / person;
LC= 20000;

! Worker Idle Cost/hour;
JW= 4;

!Export Results to Excel;
@OLE('D:\EX8THESIS.XLS', 'Part_Family', 'Worker', 'Machine', 'Freq', 'CLI') = X, W, MA,
FREQ, CLI;

ENDDATA

! Objective Function;
MIN = MACHINE_COST + MACHINE_IDLE_COST + WORKER_COST + WORKER_IDLE_COST + INTERCELL_COST
+ LIFTING_COST + MCHANGE_COST + WCHANGE_COST;

! Machine Cost + Machine Idle Cost;
MACHINE_COST = @SUM(MTC(M,T,C): F(M)*MA(M,T,C));
MACHINE_IDLE_COST = @SUM(MTC(M,T,C): JI(M)*MI(M,T,C)/R);

! Intercell Movements Cost;
INTERCELL_COST = @SUM(PJTC(P,J,T,C)|J#GE#2: H(P)*D(P,T)*V(P,J,T,C));

! Worker Cost + Worker Idle Cost;
WORKER_COST = @SUM(CT(C,T): N*U*W(C,T));
WORKER_IDLE_COST = @SUM(CT(C,T): JW*WI(C,T)/R);

! Lifting penalty cost;
LIFTING_COST = @SUM(CT(C,T): LC * W(C,T) * CLI(C,T)) ;

! Machine Capacity Change;
MCHANGE_COST = @SUM(MTC(M,T,C): IM(T)*DPM(M,T,C) + DM(T)*DNM(M,T,C) );

! Worker Capacity Change;
WCHANGE_COST = @SUM(CT(C,T): IW(T)*DPW(C,T) + DW(T)*DNW(C,T) );

! Constraint A: 1 job to 1 cell;
@FOR(PJT(P,J,T): @SUM(CELL(C): X(P,J,T,C)) = 1 );

! Constraint B: Enough machining time;
@FOR(MTC(M,T,C): MI(M,T,C) + @SUM(PJ(P,J): D(P,T)*OT(P,J,M)*X(P,J,T,C)) = MA(M,T,C)*U*R
);

! Constraint C: Cell capacity;
@FOR(CT(C,T): @SUM(MACHINE(M): MA(M,T,C)) <= S(C,T));
@FOR(CT(C,T): W(C,T) <= VW(C,T));
```

```
! Constraint D: Intercell movements;
@FOR(PJTC(P,J,T,C)|J#GE#2: X(P,J,T,C) - X(P,J-1,T,C) = V(P,J,T,C) - VN(P,J,T,C));

! Constraint E: Enough worker capacity;
@FOR(CT(C,T): WI(C,T) + @SUM(PJM(P,J,M): G(M)*D(P,T)*OT(P,J,M)*X(P,J,T,C)) = W(C,T)*U*R
);

! Constraint F: Lifting Frequency Range;
@FOR(PJ(P,J)|@SUM(MACHINE(M):OT(P,J,M))#NE#0 : L(P,J) = 1 );
@FOR(PJ(P,J)|@SUM(MACHINE(M):OT(P,J,M))#EQ#0 : L(P,J) = 0 );
@FOR(CT(C,T): FREQ(C,T) = @SUM(PJ(P,J): D(P,T)*X(P,J,T,C)*L(P,J)/U/R/W(C,T)) );
@FOR(CT(C,T): FREQ(C,T) <= 3 );

! Constraint F: Composite Lifting Index;
@FOR(PJTC(P,J,T,C) : FM(P,J,T,C) = 0.8359-(0.0893464*D(P,T)*X(P,J,T,C)*L(P,J)/U/R/W(C,T))
);
@FOR(PJTC(P,J,T,C) : STLI(P,J,T,C) =
X(P,J,T,C)*L(P,J)*LW(P)/(19.55*(0.82+4.5/LD(P))*FM(P,J,T,C)) );
@FOR(CT(C,T): INCR(C,T) = 0.25*(@SUM(PJ(P,J):STLI(P,J,T,C))- @MAX(PJ(P,J):
STLI(P,J,T,C))) / (@SUM(PJ(P,J): X(P,J,T,C)*L(P,J))-1));
@FOR(CT(C,T): CLI(C,T) = @MAX(PJ(P,J): STLI(P,J,T,C)) + INCR(C,T));
@FOR(CT(C,T): CLI(C,T) <= 1.5);

! Constraint G: Machine & Worker Balance Equations;
@FOR(MTC(M,T,C)|T#GE#2: MA(M,T,C) - MA(M,T-1,C) = DPM(M,T,C) - DNM(M,T,C));
@FOR(CT(C,T)|T#GE#2: W(C,T) - W(C,T-1) = DPW(C,T) - DNW(C,T));

! Constraint H: 0/1 Integer;
@FOR(PJTC(P,J,T,C): @BIN(X(P,J,T,C)) );
@FOR(PJTC(P,J,T,C): @BIN(V(P,J,T,C)) );
@FOR(PJTC(P,J,T,C): @BIN(VN(P,J,T,C)) );

! Constraint I: Non-zero decision variables;
@FOR(MTC(M,T,C): MA(M,T,C) >= 0 );
@FOR(MTC(M,T,C): MI(M,T,C) >= 0 );
@FOR(CT(C,T): W(C,T) >= 0 );
@FOR(CT(C,T): WI(C,T) >= 0 );
@FOR(MTC(M,T,C): DPM(M,T,C) >= 0);
@FOR(MTC(M,T,C): DNM(M,T,C) >= 0);
@FOR(CT(C,T): DPW(C,T) >= 0 );
@FOR(CT(C,T): DNW(C,T) >= 0 );

! Constraint J: General integer;
@FOR(MTC(M,T,C): @GIN(MA(M,T,C)) );
@FOR(CT(C,T): @GIN(W(C,T)) );
@FOR(MTC(M,T,C): @GIN(DPM(M,T,C)) );
@FOR(MTC(M,T,C): @GIN(DNM(M,T,C)) );
@FOR(CT(C,T): @GIN(DPW(C,T)) );
@FOR(CT(C,T): @GIN(DNW(C,T)) );

END
```

Appendix II

Genetic Algorithms:

MATLAB Source Codes

A.2.1. GA1 Source Codes (m-files)

A.2.1.1. Example 1: Main Algorithm (Ex1Thesis.m)

```
clear;
clc;

% Parts Information
P = [22000,24000,18000,17500,20000,21000,16000,19000];  % Parts Demand
LW = [9, 17, 15, 16, 11, 10, 12, 8];                    % Parts Weight
LD = [40, 45,50,60,40,45,55,35];                        % Lifting Distance

% Machines Information
M = [16000,20000,18000,12500,14000];     % Machine Procurement Cost
AT= [  .75,  .60,  .80,  .70,  .65];     % Operator Attention Needed
IC= [    4,    6,    5,    6,    3];     % Machine Idle Cost / hour

% Cells Information
C = [8,7];       % Max # of Machines
D = [6,6];       % Max # or Workers

% Processing Time of Parts on Machines (in minute)
T = [  2,   4,  0,      3,   0;
       0, 3.2,          0,    1.8,   0;
       0,      0, 2.2,   0, 3.5;
       0,      2,       0,    2.3,   0;
       3, 2.6,  0,       0,   0;
       0, 1.9, 2.1,    0, 2.4;
       0,      0, 2.5,   0, 2.7;
     2.6, 2.9,    0, 2.4,   0];


% Other Information
O = [2000,10,4,15000];       % Total Time, Worker Salary, Worker Idle Cost, Lifting Cost

% Parameter [Population Size, Max Generation, pc, pm, Number of Cells]
Parameter = [length(P),1500*length(C),.80,.08,length(C)];

tic

[FNew] = generate(Parameter,P,M,C);

for g=1:Parameter(2)

    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,M,AT,IC,C,D,T,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);

    % Generation Report
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
           end
         end

    % Update the new generation
    FNew = FM;

end

toc
```

```
% Final Report
BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

 [SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,M,AT,IC,C,D,T,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.1.2. Example 2: Main Algorithm (Ex2Thesis.m)

```
clear;
clc;

% Parts Information
P = [21000,     15000,  17000,  18000,  16000,  17000,  16500,  16000,  20000,  18500,  16000,
       18000,   19000,  17500,  17000];
LW= [17,        13,     7,      10,     9,      11,     7,      15,     6,      11,     12,
       10,      8,      9,      12];
LD= [40,        45,     45,     47,     52,     38,     48,     50,     60,     40,     45,
       55,      35,     50,     40];

% Machines Information
M = [25000,     26000,  24000,  27000,  28000,  29000,  23000,  25000,  30000,  28000];
AT= [0.75,      0.5,    0.45,   0.6,    0.5,    0.6,    0.8,    0.7,    0.65,   0.55];
IC= [4, 5,      3,      5,      4,      6,      5,      6,      3,      4];

% Cells Information
C = [7,6,6];
D = [5,5,4];

% Processing Time of Parts on Machines (in minute)
T = [  0,       0, 1.8, 1.9,    0,      2.2,    0,      0,      0,      0;
       0,       0,      0,      0,      2.4,    0,      2,      0,      0,      0;
       0,       0,      0,      0,      0,      0,      0,      2.3,    0,      2;
       0,       0,      0,      0,      1.4,    0,      2.2,    2,      0,      1.6;
       0,  3.2,    0,    3.1,    0,      0,      0,      0,      2.4,    0;
     2.6,  2.4,    0,    0,      0,      0,      0,      0,      2,      0;
       0,       0,      0,      3,      2,      0,      2,      0,      0,      0;
       0,       0, 1.7, 1.8,    0,      2.8,    0,      0,      0,      0;
     2.8,  1.3,    0,    0,      0.8,    0,      0,      0,      0,      0;
       0,       0,      0,      0,      1.8,    0,      1.8,    0,      0,      0;
       0,       0,      0,      0,      1.9,    0,      2.2,    2.1,    0,      1.5;
       0,       0,      0,      3.2,    0,      0,      1.7,    0,      0,      0;
       0,       0,  2.9, 0,     0,      0,      0,      0,      1.8,    0;
       0,       0,    0, 0,     2,      0,      0,      0,      0,      1.5;
       2,       0,      0,      0,      2.1,    0,      1.6,    0,      0,      0];

% Other Information
O = [2000,10,4,15000];

% Parameter
Parameter = [length(P),1500*length(C),.80,.08,length(C)];

tic
```

```
[FNew] = generate(Parameter,P,M,C);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,M,AT,IC,C,D,T,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
           end
         end
    FNew = FM;
end

toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,M,AT,IC,C,D,T,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.1.3. Procedure GENERATE (generate.m)

```
function [PopF] = generate(Par,P,M,C)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));

for s=1:Par(1)
    PopF(:,:,s) = randint(1,TP,[1,TC]);
end
```

A.2.1.4. Procedure EVALUATION (evaluation.m)

```
function [G,Obj_Function,Prob,Costs] = evaluation(Par,FOrg,P,LW,LD,M,AT,IC,C,D,T,O)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));
Period = 1;
```

```
for u=1:Par(1)
    for v=1:Period
        for c=1:TC
            for p=1:TP
                F(v,TP*(c-1)+p,u)=0;
                if FOrg(v,p,u) == c
                   F(v,TP*(c-1)+p,u)=1;
                   end
               end
            end
        end
    end



for s=1:Par(1)
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            Machine_Used = Machine_Used + P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            Worker_Used = Worker_Used + AT(m)*P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        end
    G(r,(TM+1)*c,s) =  ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
    for m=1:TM
```

```
        L(p,m)=0;
        if T(p,m) > 0
            L(p,m)=1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                Freq = Freq + F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s);
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
Counter = 0;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        TotalT=0;
        STLI(r,TP*(c-1)+p,s)=0;
        if sum(F(r,TP*(c-1)+1:TP*c,s)) ~= 0
           FM(p,c)=.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s) ;
           STLI(r,TP*(c-1)+p,s) = LW(1,p)*F(r,TP*(c-
1)+p,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s));
           end
        end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*(c-1)+1:TP*c,s);
    IndF = F(r,TP*(c-1)+1:TP*c,s);
    CLI(r,c,s) = 0;
    if sum(IndF) <= 1
       CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndF) >= 2
       CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndF)-1);
       end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      Counter = Counter + 1;
      B(c)=1;
      end
    end
  end


% OBJECTIVE FUNCTION
% Machine Cost
```

```
Machine_Cost(s,r) = 0;
for c=1:TC
    for m=1:TM
        Machine_Cost(s,r) = Machine_Cost(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
Machine_Idle_Cost(s,r) = 0;
for c=1:TC
    for m=1:TM
        Machine_Idle_Cost(s,r) = Machine_Idle_Cost(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) /
60;
        end
    end

% Worker Cost
Worker_Cost(s,r)=0;
for c=1:TC
    Worker_Cost(s,r) = Worker_Cost(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
Worker_Idle_Cost(s,r)=0;
for c=1:TC
    Worker_Idle_Cost(s,r) = Worker_Idle_Cost(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Lifting Cost
Lifting_Cost(s,r)=0;
for c=1:TC
    Lifting_Cost(s,r) = Lifting_Cost(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end


end   % end of period

Based_Penalty = max(Machine_Cost(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied FoL, & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
        if MinC(r,c,s) > C(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
           end
        if G(r,(TM+1)*c,s) > D(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
           end
        if FoL(r,c,s) > 3
           Pen(s) = Pen(s) + Based_Penalty * 100 * (FoL(r,c,s) - 3);
           end
        if CLI(r,c,s) > 1.5
           Pen(s) = Pen(s) + Based_Penalty * 100 * (CLI(r,c,s) - 1.5);
           end
       end
    end


Costs(s,1) = sum(Machine_Cost(s,:)) + sum(Worker_Cost(s,:)) + sum(Machine_Idle_Cost(s,:))
+ sum(Worker_Idle_Cost(s,:)) + sum(Lifting_Cost(s,:)) ;
Costs(s,2) = Pen(s);
Obj_Function(s) = Costs(s,1) + Costs(s,2);

end   % end of pop size

Total_Fitness=0;
Dim=min(Obj_Function);
for s=1:Par(1)
```

```
    Fitness_Value(s,1) = (Dim/1000) / ((Dim/1000)+Obj_Function(s)-Dim);
    Total_Fitness = Total_Fitness + Fitness_Value(s,1);
    end

for s=1:Par(1)
    Prob(s,1) = Fitness_Value(s,1)/Total_Fitness;
    end

Prob(1,2)=Prob(1,1);
for s=2:Par(1)
    Prob(s,2) = Prob(s-1,2) + Prob(s,1);
    end
```

A.2.1.5. Procedure SELECTION (selection.m)

```
function [PopF,Ra] = selection(Par,Prob,F)

for s=1:Par(1)
    Ra(s) = rand;
    for t=2:Par(1)
        if Ra(s) <= Prob(1,2)
            PopF(:,:,s) = F(:,:,1);
            end
        if (Prob(t-1,2) < Ra(s)) & (Ra(s) <= Prob(t,2))
            PopF(:,:,s) = F(:,:,t);
            end
        end
    end
```

A.2.1.6. Procedure CROSSOVER (crossover.m)

```
function [PopF,RaC] = crossover(Par,F)

PopF=F;

Period=1;

Crossover=[0,0];
while mod(length(Crossover),2) ~= 1
    Crossover=0;
    for s=1:Par(1)
        RaC(s) = rand;
        if RaC(s) <= Par(3)
            Crossover = [Crossover,s];
           end
        end
    end

Crossover(1)=[];

for i=1:length(Crossover)/2
    CPair(i,1) = Crossover(2*i-1);
    CPair(i,2) = Crossover(2*i);
    for q=1:Period
        CPair(i,2+q) = randint(1,1,[1,Period]);
        end
    end


Choice=randint(1,length(Crossover)/2,[1,3]);
```

```
for i=1:length(Crossover)/2

% 1 point Crossover
if Choice(i) == 1
    Point1F = randint(1,1,[1,length(F(1,:,1))-1]);
    for q=1:Period
        for v=Point1F+1:length(F(1,:,1))
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 2 points Crossover
if Choice(i) == 2
    Point2F=randint(1,2,[1,length(F(1,:,1))-1]);
    Point2F=sort(Point2F);
    for q=1:Period
        for v=Point2F(1)+1:Point2F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 3 points Crossover
if Choice(i) == 3
    Point3F=randint(1,3,[1,length(F(1,:,1))-1]);
    Point3F=sort(Point3F);
    for q=1:Period
        for v=Point3F(1)+1:Point3F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        for w=Point3F(3)+1:length(F(1,:,1))
            PopF(CPair(i,2+q),w,CPair(i,1)) = F(CPair(i,2+q),w,CPair(i,2));
            PopF(CPair(i,2+q),w,CPair(i,2)) = F(CPair(i,2+q),w,CPair(i,1));
            end
        end
    end

end
```

A.2.1.7. Procedure MUTATION (mutation.m)

```
function [PopFM,Choice] = mutation(Par,F)

TC=Par(5);

PopFM=F;

Period=1;

Mutation=[0,0];
for s=1:Par(1)
    for v=1:Period
        RaM(s,v) = rand;
        if RaM(s,v) <= Par(4)
            Mutation = [Mutation;s,v];
            end
        end
    end
 Mutation(1,:)=[];
```

```
Choice=randint(1,length(Mutation(:,1)),[1,3]);

for u=1:length(Mutation(:,1));

% Self Mutation
if Choice(u) == 1
    TotalPoint = randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF = randint(1,TotalPoint,[1,length(F(1,:,1))]);
    A = randint(1,TotalPoint,[1,TC]);
    for i=1:TotalPoint
        while A(i) == PopFM(Mutation(u,2),PointF(i),Mutation(u,1))
            A(i)=randint(1,1,[1,TC]);
            end
        PopFM(Mutation(u,2),PointF(i),Mutation(u,1)) = A(i);
        end
    end


% Switch Mutation
if Choice(u) == 2
    P=randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        PopFM(Mutation(u,2),PointF(i,2),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,1),Mutation(u,1));
        end
    end

% Copy Mutation
if Choice(u) == 3
    P=randint(1,1,[1,round(length(F(1,:,1))/2)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        end
    end

end
```

A.2.1.8. Procedure SUMMARY (summary.m)

```
function [SolF,G,System_Cost,Costs,MC,WC,MIC,WIC,LC,MinC,FoL,STLI,CLI] =
summary(FOrg,P,LW,LD,M,AT,IC,C,D,T,O)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));
Period = 1;


SolF=FOrg;

for v=1:Period
        for c=1:TC
            for p=1:TP
                F(v,TP*(c-1)+p,1)=0;
```

```
                if FOrg(v,p,1) == c
                   F(v,TP*(c-1)+p,1)=1;
                   end
                end
            end
        end

s=1;
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            Machine_Used = Machine_Used + P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            Worker_Used = Worker_Used + AT(m)*P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        end
    G(r,(TM+1)*c,s) =  ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
    for m=1:TM
        L(p,m)=0;
        if T(p,m) > 0
            L(p,m)=1;
            end
        end
    end
for c=1:TC
    A=1;
```

```
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                Freq = Freq + F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s);
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
Counter = 0;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        TotalT=0;
        STLI(r,TP*(c-1)+p,s)=0;
        if sum(F(r,TP*(c-1)+1:TP*c,s)) ~= 0
           FM(p,c)=.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s) ;
            STLI(r,TP*(c-1)+p,s) = LW(1,p)*F(r,TP*(c-
1)+p,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s));
            end
        end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*(c-1)+1:TP*c,s);
    IndF = F(r,TP*(c-1)+1:TP*c,s);
    CLI(r,c,s) = 0;
    if sum(IndF) <= 1
       CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndF) >= 2
       CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndF)-1);
       end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      Counter = Counter + 1;
      B(c)=1;
      end
    end
  end



% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
```

```
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end

end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied FoL, & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
        if MinC(r,c,s) > C(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
           end
        if G(r,(TM+1)*c,s) > D(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
           end
        if FoL(r,c,s) > 3
           Pen(s) = Pen(s) + Based_Penalty * 100 * (FoL(r,c,s) - 3);
           end
        if CLI(r,c,s) > 1.5
           Pen(s) = Pen(s) + Based_Penalty * 100 * (CLI(r,c,s) - 1.5);
           end
       end
    end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MIC(s,:)) + sum(WIC(s,:)) + sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);
```

A.2.2. GA2 Source Codes (m-files)

A.2.2.1. Example 3: Main Algorithm (Ex3Thesis.m)

```
clear;
clc;

P = [22000,    24000,  30000,  27000,  20000,  21000,  24000,  19000,  27000];
LW= [10,       9,      17,     14,     12,     6,      8,      20,     5];
LD= [45,       50,     41,     60,     67,     45,     35,     40,     40];
IM= [0.3,      0.5,    0.4,    0.35,   0.25,   0.4,    0.3,    0.35,   0.4];
```

```
% Machines Information
M = [16000,    20000,  18000,  12500,  14000,  17000];
AT= [0.75,     0.6,    0.8,    0.7,    0.65,   0.55];
IC= [4, 6,     5,      6,      3,      4];

% Cells Information
C = [8,10];
D = [6,7];

% Processing Time of Jobs on Machines
T =  [  2,     0,      0,      0,      0,      0;
        0,     4,      0,      0,      0,      0;
        0,     0,      0,      3,      0,      0;
        0,     0,      0,      0,      0,      0;
        0,     0,      0,      0,      0,      1.7;
        0,     0,      0,      0,      1.8,    0;
        0,     3.2,    0,      0,      0,      0;
        0,     0,      0,      0,      0,      0;
        0,     0,      2.2,    0,      0,      0;
        0,     0,      0,      0,      3.5,    0;
        0,     0,      0,      0,      0,      0;
        0,     0,      0,      0,      0,      0;
        0,     0,      0,      2.3,    0,      0;
        0,     2,      0,      0,      0,      0;
        0,     0,      0,      0,      0,      3;
        0,     0,      0,      0,      0,      0;
        0,     2.6,    0,      0,      0,      0;
        0,     0,      0,      0,      0,      2.8;
        0,     0,      0,      0,      0,      0;
        0,     0,      0,      0,      0,      0;
        0,     0,      0,      0,      2.4,    0;
        0,     0,      2.1,    0,      0,      0;
        0,     1.9,    0,      0,      0,      0;
        0,     0,      0,      0,      0,      0;
        0,     2.9,    0,      0,      0,      0;
        2.6,   0,      0,      0,      0,      0;
        0,     0,      0,      2.4,    0,      0;
        0,     0,      0,      0,      0,      2.2;
        0,     0,      0,      0,      2.2,    0;
        0,     0,      2,      0,      0,      0;
        0,     0,      0,      0,      0,      2;
        0,     0,      0,      0,      0,      0;
        0,     0,      2.8,    0,      0,      0;
        0,     3.2,    0,      0,      0,      0;
        0,     0,      0,      2,      0,      0;
        3,     0,      0,      0,      0,      0];


% Other Information
O = [2000,10,4,15000];

% Parameter
Parameter = [length(P),1500*length(C),.80,.08,length(C)];

tic

[FNew] = generate(Parameter,P,M,C,T);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,IM,M,AT,IC,C,D,T,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
```

```
            BestGGen(:,:,g) = G(:,:,h);
            ObjGen(g) = Obj(h);
        end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,IMC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,IM,M,AT,IC,C,D,T,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.2.2. Example 4: Main Algorithm (Ex4Thesis.m)

```
clear;
clc;

P = [26000,    28000,  17000,  27000,  29000,  19000,  25000,  24000,  16000,  26000,  16000,
     18000,    24000,  22000,  28000,  20000,  30000,  24000,  18000,  19000];
LW= [15,       9,      11,     7,      12,     17,     10,     8,      9,      12,     11,
     17,       15,     6,      11,     13,     14,     21,     10,     9];
LD= [57,       41,     49,     54,     40,     45,     52,     36,     51,     42,     40,
     66,       57,     52,     43,     41,     45,     30,     37,     48];
IM= [0.3,      0.5,    0.4,    0.35,   0.25,   0.4,    0.45,   0.3,    0.35,   0.4,    0.3,
     0.5,      0.4,    0.35,   0.25,   0.35,   0.25,   0.4,    0.45,   0.5];

% Machines Information
M = [25000,    26000,  23000,  15000,  15000,  24000,  20000,  18000,  17000,  16000,  22000,
     21000];
AT= [0.75,     0.6,    0.75,   0.7,    0.65,   0.55,   0.75,   0.7,    0.5,    0.55,   0.5,
     0.55];
IC= [4, 6,     5,      6,      3,      4,      4,      6,      5,      6,      5,      3];


% Cells Information
C = [7, 14, 12];
D = [7, 10, 10];

% Processing Time of Jobs on Machines
T = [0,0,1.9,0,0,0,0,0,0,0,0,0;       %Part1
0,0,0,0,0,1.8,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,2.8,0,0,0,0,0,0;    %Part2
0,2.3,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,2.2,0,0,0,0,0,0,0;    %Part3
0,2.2,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,1.9,0,0,0;
```

```
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,2.1,0,0;     %Part4
0,2.7,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,3.1,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,1.6,0,0,0,0,0,0,0;     %Part5
0,0,1.7,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,1.8,0;
0,0,0,0,0,0,0,0,0,0,0,0;
2.9,0,0,0,0,0,0,0,0,0,0,0;     %Part6
0,0,0,0,1.8,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,2.5,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,1.4,0,0,0,0,0;     %Part7
0,0,0,0,0,0,0,0,0,0,1.8,0;
0,0,0,1.7,0,0,0,0,0,0,0,0;
2.9,0,0,0,0,0,0,0,0,0,0,0;
0,0,1.8,0,0,0,0,0,0,0,0,0;     %Part8
0,0,0,0,0,0,0,0,0,0,0,2.8;
0,0,0,0,0,0,2.9,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,2.8,0,0,0,0,0;     %Part9
0,0,0,0,0,0,0,0,0,0,2.9,0,0;
0,1.7,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
1.5,0,0,0,0,0,0,0,0,0,0,0;     %Part10
0,0,0,0,0,0,0,0,0,1.8,0,0;
0,0,0,1.8,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,1.7,0,0,0,0,0,0,0;     %Part11
0,0,0,0,0,0,0,0,2.1,0,0,0;
0,0,1.5,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,1.6,0,0,0,0,0,0,0,0;     %Part12
2,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,2.4,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,2.1,0,0,0,0,0,0,0,0,0;     %Part13
0,0,0,0,0,2.3,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,1.9,0,0,0,0,0,0,0,0;     %Part14
0,2.5,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,2,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,2.5,0,0,0,0,0,0,0,0,0;     %Part15
0,0,0,0,0,0,0,0,0,0,0,3.2;
0,0,0,0,0,0,3.1,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
2.3,0,0,0,0,0,0,0,0,0,0,0;     %Part16
0,0,0,0,2.8,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,2.9,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,2.1,0,0,0,0,0,0,0,0;     %Part17
0,0,2.9,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,2.7,0;
0,0,0,0,0,0,0,0,0,0,0,0;
1.3,0,0,0,0,0,0,0,0,0,0,0;     %Part18
0,0,0,1.7,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
1.7,0,0,0,0,0,0,0,0,0,0,0;     %Part19
0,0,0,2,0,0,0,0,0,0,0,0;
0,0,0,0,0,3,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,3.2,0,0,0,0,0,0;     %Part20
0,0,0,0,0,0,0,0,0,0,1.6,0,0;
0,2.1,0,0,0,0,0,0,0,0,0,0;
0,0,0,0,0,0,0,0,0,0,0,0];
```

```
% Other Information
O = [2000,10,4,15000];

% Parameter
Parameter = [length(P),1500*length(C),.80,.08,length(C)];

tic

[FNew] = generate(Parameter,P,M,C,T);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,IM,M,AT,IC,C,D,T,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
       end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,IMC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,IM,M,AT,IC,C,D,T,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.2.3. Procedure GENERATE (generate.m)

```
function [PopF] = generate(Par,P,M,C,T)

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));
Period = 1;

for s=1:Par(1)
    PopF(:,:,s) = randint(Period,TP*TJ,[1,TC]);
end
```

A.2.2.4. Procedure EVALUATION (evaluation.m)

```
function [G,System_Cost,Prob,Costs] = evaluation(Par,FOrg,P,LW,LD,IM,M,AT,IC,C,D,T,O)

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));

Period = 1;

for u=1:Par(1)
  for v=1:Period
        for c=1:TC
            for p=1:TP
                for j=1:TJ
                    F(v,TP*TJ*(c-1)+TJ*(p-1)+j,u)=0;
                    if FOrg(v,TJ*(p-1)+j,u) == c
                        F(v,TP*TJ*(c-1)+TJ*(p-1)+j,u)=1;
                        end
                    end
                end
            end
        end
    end


for s=1:Par(1)
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            for j=1:TJ
                Machine_Used = Machine_Used + P(r,p)*T(TJ*(p-1)+j,m)*F(r,TP*TJ*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            for j=1:TJ
                Worker_Used = Worker_Used + AT(m)*P(r,p)*T(TJ*(p-1)+j,m)*F(r,(TP*TJ)*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        end
    G(r,(TM+1)*c,s) = ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end
```

```
% Intercell Moves
for c=1:TC
    for p=1:TP
        for j=2:TJ
            V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) -
F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j-1,s);
            if  (V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) == -1 )
                V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = 0;
                end
            end
        end
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end

% Frequency of Lifting
for p=1:TP
    for j=1:TJ
        L(p,j)=0;
        if sum(T(TJ*(p-1)+j,:)) > 0
            L(p,j) = 1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                for j=1:TJ
                    Freq = Freq + F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*P(r,p)*L(p,j)/O(1,1)/60/G(r,(TM+1)*c,s);
                    end
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        for j=1:TJ
        TotalT=0;
        STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
```

```
        if (sum(T(TJ*(p-1)+j,:)) ~= 0) & (G(r,(TM+1)*c,s) ~=0 )
           ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = L(p,j)*F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
           FM(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = (.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = LW(1,p)*ASG(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           end
       end
       end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    IndASG=ASG(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    CLI(r,c,s) = 0;
    if sum(IndASG) <=1
        CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndASG) >= 2
        CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndASG)-1);
        end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      B(c)=1;
      end
    end
  end


% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Intercell Movements Cost
IMC(s,r)=0;
for c=1:TC
    for p=1:TP
        for j=2:TJ
            IMC(s,r) = IMC(s,r) + P(r,p)*IM(1,p)*V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
            end
        end
    end
```

```
% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end

end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied Freq of Lifting & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
        if MinC(r,c,s) > C(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
           end
        if G(r,(TM+1)*c,s) > D(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
           end
        if FoL(r,c,s) > 3
           Pen(s) = Pen(s) + Based_Penalty*100* (FoL(r,c,s) - 3);
           end
        if CLI(r,c,s) > 1.5
           Pen(s) = Pen(s) + Based_Penalty*100* (CLI(r,c,s) - 1.5);
           end
       end
    end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MIC(s,:)) + sum(WIC(s,:)) + sum(IMC(s,:))
+ sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);

end  % end of pop size

Total_Fitness=0;
Dim=min(System_Cost);

for s=1:Par(1)
    Fitness_Value(s,1) = (Dim/1000) / ((Dim/1000)+System_Cost(s)-Dim);
    Total_Fitness = Total_Fitness + Fitness_Value(s,1);
    end

for s=1:Par(1)
    Prob(s,1) = Fitness_Value(s,1)/Total_Fitness;
    end

Prob(1,2)=Prob(1,1);
for s=2:Par(1)
    Prob(s,2) = Prob(s-1,2) + Prob(s,1);
    end
```

A.2.2.5. Procedure SELECTION (selection.m)

```
function [PopF,Ra] = selection(Par,Prob,F)

for s=1:Par(1)
    Ra(s) = rand;
    for t=2:Par(1)
        if Ra(s) <= Prob(1,2)
            PopF(:,:,s) = F(:,:,1);
            end
        if (Prob(t-1,2) < Ra(s)) & (Ra(s) <= Prob(t,2))
            PopF(:,:,s) = F(:,:,t);
```

```
            end
        end
    end
```

A.2.2.6. Procedure CROSSOVER (crossover.m)

```
function [PopF,RaC] = crossover(Par,F)

PopF=F;

Period=1;

Crossover=[0,0];
while mod(length(Crossover),2) ~= 1
    Crossover=0;
    for s=1:Par(1)
        RaC(s) = rand;
        if RaC(s) <= Par(3)
            Crossover = [Crossover,s];
           end
        end
    end

Crossover(1)=[];

for i=1:length(Crossover)/2
    CPair(i,1) = Crossover(2*i-1);
    CPair(i,2) = Crossover(2*i);
    for q=1:Period
        CPair(i,2+q) = randint(1,1,[1,Period]);
        end
    end


Choice=randint(1,length(Crossover)/2,[1,3]);

for i=1:length(Crossover)/2

% 1 point Crossover
if Choice(i) == 1
    Point1F = randint(1,1,[1,length(F(1,:,1))-1]);
    for q=1:Period
        for v=Point1F+1:length(F(1,:,1))
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 2 points Crossover
if Choice(i) == 2
    Point2F=randint(1,2,[1,length(F(1,:,1))-1]);
    Point2F=sort(Point2F);
    for q=1:Period
        for v=Point2F(1)+1:Point2F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 3 points Crossover
if Choice(i) == 3
    Point3F=randint(1,3,[1,length(F(1,:,1))-1]);
    Point3F=sort(Point3F);
```

```
    for q=1:Period
        for v=Point3F(1)+1:Point3F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        for w=Point3F(3)+1:length(F(1,:,1))
            PopF(CPair(i,2+q),w,CPair(i,1)) = F(CPair(i,2+q),w,CPair(i,2));
            PopF(CPair(i,2+q),w,CPair(i,2)) = F(CPair(i,2+q),w,CPair(i,1));
            end
        end
    end

end
```

A.2.2.7. Procedure MUTATION (mutation.m)

```
function [PopFM,Choice] = mutation(Par,F)

TC=Par(5);

PopFM=F;

Period=1;

Mutation=[0,0];
for s=1:Par(1)
    for v=1:Period
        RaM(s,v) = rand;
        if RaM(s,v) <= Par(4)
            Mutation = [Mutation;s,v];
            end
        end
    end
 Mutation(1,:)=[];


Choice=randint(1,length(Mutation(:,1)),[1,3]);

for u=1:length(Mutation(:,1));

% Self Mutation
if Choice(u) == 1
    TotalPoint = randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF = randint(1,TotalPoint,[1,length(F(1,:,1))]);
    A = randint(1,TotalPoint,[1,TC]);
    for i=1:TotalPoint
        while A(i) == PopFM(Mutation(u,2),PointF(i),Mutation(u,1))
            A(i)=randint(1,1,[1,TC]);
            end
        PopFM(Mutation(u,2),PointF(i),Mutation(u,1)) = A(i);
        end
    end


% Switch Mutation
if Choice(u) == 2
    P=randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
```

```
        PopFM(Mutation(u,2),PointF(i,2),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,1),Mutation(u,1));
        end
    end

% Copy Mutation
if Choice(u) == 3
    P=randint(1,1,[1,round(length(F(1,:,1))/2)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        end
    end


end
```

A.2.2.8. Procedure SUMMARY (summary.m)

```
function [SolF,G,System_Cost,Costs,MC,WC,MIC,WIC,IMC,LC,MinC,FoL,STLI,CLI] =
summary(FOrg,P,LW,LD,IM,M,AT,IC,C,D,T,O)

SolF=FOrg;

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));

Period = 1;

    for v=1:Period
        for c=1:TC
            for p=1:TP
                for j=1:TJ
                    F(v,TP*TJ*(c-1)+TJ*(p-1)+j,1)=0;
                    if FOrg(v,TJ*(p-1)+j,1) == c
                        F(v,TP*TJ*(c-1)+TJ*(p-1)+j,1)=1;
                        end
                    end
                end
            end
        end



s=1;
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            for j=1:TJ
                Machine_Used = Machine_Used + P(r,p)*T(TJ*(p-1)+j,m)*F(r,TP*TJ*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
```

```
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            for j=1:TJ
                Worker_Used = Worker_Used + AT(m)*P(r,p)*T(TJ*(p-1)+j,m)*F(r,(TP*TJ)*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        end
    G(r,(TM+1)*c,s) = ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Intercell Moves
for c=1:TC
    for p=1:TP
        for j=2:TJ
            V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) -
F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j-1,s);
            if  (V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) == -1 )
                V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = 0;
                end
            end
        end
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
    for j=1:TJ
        L(p,j)=0;
        if sum(T(TJ*(p-1)+j,:)) > 0
            L(p,j) = 1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
```

```
            Freq = 0;
            for p=1:TP
                for j=1:TJ
                    Freq = Freq + F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*P(r,p)*L(p,j)/O(1,1)/60/G(r,(TM+1)*c,s);
                end
            end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
        end
    end


% Composite Lifting Index
B(1:TC)=1;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        for j=1:TJ
        TotalT=0;
        STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        if (sum(T(TJ*(p-1)+j,:)) ~= 0) & (G(r,(TM+1)*c,s) ~=0 )
           ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = L(p,j)*F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
           FM(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = (.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = LW(1,p)*ASG(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           end
       end
       end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    IndASG=ASG(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    CLI(r,c,s) = 0;
    if sum(IndASG) <=1
        CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndASG) >= 2
        CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndASG)-1);
        end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      B(c)=1;
      end
    end
  end


% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
```

```
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Intercell Movements Cost
IMC(s,r)=0;
for c=1:TC
    for p=1:TP
        for j=2:TJ
            IMC(s,r) = IMC(s,r) + P(r,p)*IM(1,p)*V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
            end
        end
    end

% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end

end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied Freq of Lifting & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
         if MinC(r,c,s) > C(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
            end
         if G(r,(TM+1)*c,s) > D(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
            end
         if FoL(r,c,s) > 3
            Pen(s) = Pen(s) + Based_Penalty*100* (FoL(r,c,s) - 3);
            end
         if CLI(r,c,s) > 1.5
            Pen(s) = Pen(s) + Based_Penalty*100* (CLI(r,c,s) - 1.5);
            end
        end
    end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MIC(s,:)) + sum(WIC(s,:)) + sum(IMC(s,:))
+ sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);
```

A.2.3. GA3 Source Codes (m-files)

A.2.3.1. Example 5: Main Algorithm (Ex5Thesis.m)

```
clear;
clc;

% Parts Information
P = [21000,    19000, 20000, 20000, 22000, 23000, 22000, 19000, 18000, 20000;
     22000,    16000, 18000, 21000, 23000, 9000,  19000, 20000, 20000, 9000;
     21000,    13000, 14000, 18000, 19000, 6000,  15000, 21000, 15000, 7000];

LW= [12,       9,     11,    7,     14,    16,    10,    8,     9,     12];
LD= [37,       41,    49,    54,    40,    45,    52,    36,    51,    42];


% Machines Information
M = [18000,    20000, 22000, 24000, 23000, 19000, 20000];
AT= [0.3,      0.4,   0.35,  0.55,  0.5,   0.65,  0.6];
IC= [4, 6,     5,     6,     3,     4,     4];

% Cells Information
C = [6, 6;
     5, 7;
     5, 6];

D = [4, 5;
     3, 4;
     3, 3];


% Parts Operations Sequence
T =    [  2,   0, 1.9,   0, 1.4,   0,   0;   % Part1;
              0,   0,   0, 1.7,   0, 2.2, 1.9;   % Part2;
              0,   0,   0, 2.1,   0,   0,   2;   % Part3;
              0, 1.8,   0, 2.3,   0,   0,   0;   % Part4;
              0, 1.5,   0, 2.2,   0,   0,   0;   % Part5;
              0, 2.5,   0,   0, 1.9,   0, 1.8;   % Part6;
              0,   0,   0,   2,   0, 1.7,   0;  % Part7;
            2.1,   0, 2.4,   0,   0,   0,   0;  % Part8;
            1.7,   0,   0, 2.4,   0,   0,   0;   % Part9;
              0, 2.7,   0,   0, 1.8, 1.5,   0]; % Part10;

% Machine & Worker Change
MCC = [5000,4000;
       6000,5000;
       3000,5000];
WCC = [600,600;
       400,500;
       700,300];

% Other Information
O = [2000,10,4,20000];

% Parameter
Parameter = [length(P(1,:)),1500*length(C(1,:)),.80,.08,length(C(1,:))];

tic

[FNew] = generate(Parameter,P,M,C);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
```

```
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
       end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.3.2. Example 6: Main Algorithm (Ex6Thesis.m)

```
clear;
clc;

% Parts Information
P = [26000,    28000, 17000, 27000, 29000,   11000, 25000, 24000, 16000,   26000, 16000,
18000, 24000,  22000, 28000, 20000, 30000,   24000, 18000, 19000;
     22000,    16000, 18000, 21000, 23000,  9000, 19000, 20000, 20000,  9000, 13000,
15000, 10000, 16000, 14000, 22000, 24000, 20000, 19000, 22000;
     21000, 13000, 14000, 18000, 19000,  6000, 15000, 21000, 15000, 17000,  9000, 25000,
9000,  11000,  8000, 12000, 20000, 17000, 25000, 25000;
     19000,  9000,    8000, 17000, 15000,  9000, 19000, 20000, 20000, 19000, 15000,
16000, 12000,   8000, 16000, 12000, 28000, 12000, 17000, 20000];
LW= [15,  9, 11,  7, 12, 17, 10,  8,  9, 12, 11, 17, 15,  6, 11, 13, 14, 21, 10,  9];
LD= [57,       41,     49,     54,     40,     45,     52,     36,     51,     42,     40,
       66,     57,     52,     43,     41,     45,     30,     37,     48];

% Machines Information
M = [25000, 26000, 23000, 15000, 15000, 24000, 20000, 18000, 17000, 16000, 22000, 21000];
AT= [ 0.75,   0.6,    0.75,  0.7,  0.65,  0.55,    0.75,  0.7,   0.5,  0.55,   0.5,
0.55];
IC= [    4,        6,     5,     6,     3,        4,     4,     6,     5,        6,     5,
3];

% Cells Information
C = [15,       12,     12,
15,     11,     12,
14,     11,     12,
12,     10,     12];

D = [8, 9, 9;
     9, 8, 9;
     9, 8, 8;
```

```
     8, 7, 7];


% Parts Operations Sequence
T =     [  0,    0,      1.9,    0,      0,      1.8,    0,      0,      2.1,    0,      0,
        0;
           0,    1.2,    0,      0,      0,      2.8,    0,      2.4,    0,      0,      3,
        0;
0,      0,      0,      2.1,    2.2,    0,      0,      0,      0,      0,      0,      0;
0,      2.1,    0,      0,      0,      0,      0,      0,      2.3,    2.1,    0,      0;
0,      1.2,    0,      0,      1.6,    0,      0,      0,      0,      1.9,    0,      0;
2.9,    0,      0,      0,      2.6,    0,      0,      0,      0,      0,      0,      0;
0,      1.5,    0,      0,      0,      0,      1.4,    0,      0,      0,      0,      3.2;
0,      0,      1.8,    0,      0,      0,      0,      2.2,    0,      2.5,    0,      0;
0,      0,      0,      0,      0,      0,      2.8,    0,      1.8,    0,      1.7,    2.1;
1.5,    0,      0,      0,      0,      0,      2.3,    2.7,    0,      0,      1.2,    0;
0,      0,      0,      1.2,    1.7,    0,      1.3,    0,      1.8,    0,      0,      0;
0,      2.1,    0,      1.6,    0,      0,      0,      0,      0,      0,      0,      1.3;
0,      0,      2.1,    0,      0,      0,      0,      0,      0,      2.4,    1.5,    0;
0,      0,      0,      1.9,    0,      0,      0,      0,      0,      0,      0,      2.1;
0,      1.7,    2.5,    0,      0,      2.5,    0,      0,      0,      0,      0,      0;
2.3,    0,      1.3,    0,      0,      0,      0,      1.2,    0,      0,      3.2,    0;
0,      0,      0,      2.1,    0,      0,      2.2,    0,      2,      0,      0,      0;
1.3,    1.4,    0,      1.6,    0,      0,      0,      2.2,    0,      0,      0,      0;
1.7,    0,      0,      0,      0,      0,      0,      0,      0,      3.2,    0,      0;
0,      0,      1.5,    0,      2,      3.2,    0,      0,      0,      0,      0,      1.8];


% Machine & Worker Change Cost
MCC = [4000, 5000;
       6000, 4000;
       3000, 5000;
       4000, 4500];
WCC = [600, 600;
       400, 500;
       700, 300,
       650, 400];

% Other Information
O = [2000,10,4,20000];

% Parameter
Parameter = [length(P(1,:)),1500*length(C(1,:)),.80,.08,length(C(1,:))];

tic

[FNew] = generate(Parameter,P,M,C);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
       end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
```

```
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.3.3. Procedure GENERATE (generate.m)

```
function [PopF] = generate(Par,P,M,C)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));
Period = length(P(:,1));

for s=1:Par(1)
    PopF(:,:,s) = randint(Period,TP,[1,TC]);
end
```

A.2.3.4. Procedure EVALUATION (evaluation.m)

```
function [G,Obj_Function,Prob,Costs] =
evaluation(Par,FOrg,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));
Period = length(P(:,1));


for u=1:Par(1)
    for v=1:Period
        for c=1:TC
            for p=1:TP
                F(v,TP*(c-1)+p,u)=0;
                if FOrg(v,p,u) == c
                   F(v,TP*(c-1)+p,u)=1;
                   end
               end
            end
        end
    end


for s=1:Par(1)
for r=1:Period

% CONSTRAINTS
```

```
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            Machine_Used = Machine_Used + P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            Worker_Used = Worker_Used + AT(m)*P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        end
    G(r,(TM+1)*c,s) =  ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end

% Frequency of Lifting
for p=1:TP
    for m=1:TM
        L(p,m)=0;
        if T(p,m) > 0
            L(p,m)=1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                Freq = Freq + F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s);
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
```

```
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
        end
    end


% Composite Lifting Index
B(1:TC)=1;
Counter = 0;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        TotalT=0;
        STLI(r,TP*(c-1)+p,s)=0;
        if sum(F(r,TP*(c-1)+1:TP*c,s)) ~= 0
           FM(p,c)=.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s) ;
           STLI(r,TP*(c-1)+p,s) = LW(1,p)*F(r,TP*(c-
1)+p,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s));
           end
        end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*(c-1)+1:TP*c,s);
    IndF = F(r,TP*(c-1)+1:TP*c,s);
    CLI(r,c,s) = 0;
    if sum(IndF) <= 1
       CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndF) >= 2
       CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndF)-1);
       end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      Counter = Counter + 1;
      B(c)=1;
      end
    end
  end



% OBJECTIVE FUNCTION
% Machine Cost
Machine_Cost(s,r) = 0;
for c=1:TC
    for m=1:TM
        Machine_Cost(s,r) = Machine_Cost(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
Machine_Idle_Cost(s,r) = 0;
for c=1:TC
    for m=1:TM
        Machine_Idle_Cost(s,r) = Machine_Idle_Cost(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) /
60;
        end
    end

% Worker Cost
Worker_Cost(s,r)=0;
for c=1:TC
    Worker_Cost(s,r) = Worker_Cost(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
```

```
    end

% Worker Idle Cost
Worker_Idle_Cost(s,r)=0;
for c=1:TC
    Worker_Idle_Cost(s,r) = Worker_Idle_Cost(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Lifting Cost
Lifting_Cost(s,r)=0;
for c=1:TC
    Lifting_Cost(s,r) = Lifting_Cost(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end


% Machine Relocation Cost
Machine_Change_Cost(s,r) = 0;
for c=1:TC
    for m=1:TM
        if DM(r,TM*(c-1)+m,s) < 0
            Machine_Change_Cost(s,r) = Machine_Change_Cost(s,r) + MCC(r,2) *
abs(DM(r,TM*(c-1)+m,s));
            end
        if DM(r,TM*(c-1)+m,s) > 0
            Machine_Change_Cost(s,r) = Machine_Change_Cost(s,r) + MCC(r,1) *
abs(DM(r,TM*(c-1)+m,s));
            end
        end
    end

% Worker Relocation Cost
Worker_Change_Cost(s,r) = 0;
for c=1:TC
    if DW(r,c,s) < 0
       Worker_Change_Cost(s,r) = Worker_Change_Cost(s,r) + WCC(r,2) * abs(DW(r,c,s));
       end
    if DW(r,c,s) > 0
       Worker_Change_Cost(s,r) = Worker_Change_Cost(s,r) + WCC(r,1) * abs(DW(r,c,s));
       end
    end


end   % end of period

Based_Penalty = max(Machine_Cost(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied FoL, & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
        if MinC(r,c,s) > C(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
           end
        if G(r,(TM+1)*c,s) > D(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
           end
        if FoL(r,c,s) > 3
           Pen(s) = Pen(s) + Based_Penalty * 100 * (FoL(r,c,s) - 3);
           end
        if CLI(r,c,s) > 1.5
           Pen(s) = Pen(s) + Based_Penalty * 100 * (CLI(r,c,s) - 1.5);
           end
       end
    end


Costs(s,1) = sum(Machine_Cost(s,:)) + sum(Worker_Cost(s,:)) +
sum(Machine_Change_Cost(s,:)) + sum(Worker_Change_Cost(s,:)) +
sum(Machine_Idle_Cost(s,:)) + sum(Worker_Idle_Cost(s,:)) + sum(Lifting_Cost(s,:)) ;
```

```
Costs(s,2) = Pen(s);
Obj_Function(s) = Costs(s,1) + Costs(s,2);

end   % end of pop size

Total_Fitness=0;
Dim=min(Obj_Function);
for s=1:Par(1)
    Fitness_Value(s,1) = (Dim/1000) / ((Dim/1000)+Obj_Function(s)-Dim);
    Total_Fitness = Total_Fitness + Fitness_Value(s,1);
    end

for s=1:Par(1)
    Prob(s,1) = Fitness_Value(s,1)/Total_Fitness;
    end

Prob(1,2)=Prob(1,1);
for s=2:Par(1)
    Prob(s,2) = Prob(s-1,2) + Prob(s,1);
    end
```

A.2.3.5. Procedure SELECTION (selection.m)

```
function [PopF,Ra] = selection(Par,Prob,F)

for s=1:Par(1)
    Ra(s) = rand;
    for t=2:Par(1)
        if Ra(s) <= Prob(1,2)
            PopF(:,:,s) = F(:,:,1);
            end
        if (Prob(t-1,2) < Ra(s)) & (Ra(s) <= Prob(t,2))
            PopF(:,:,s) = F(:,:,t);
            end
        end
    end
```

A.2.3.6. Procedure CROSSOVER (crossover.m)

```
function [PopF,RaC] = crossover(Par,F)

PopF=F;

Period=length(F(:,1,1));

Crossover=[0,0];
while mod(length(Crossover),2) ~= 1
    Crossover=0;
    for s=1:Par(1)
        RaC(s) = rand;
        if RaC(s) <= Par(3)
            Crossover = [Crossover,s];
           end
        end
    end

Crossover(1)=[];

for i=1:length(Crossover)/2
    CPair(i,1) = Crossover(2*i-1);
    CPair(i,2) = Crossover(2*i);
```

```
    for q=1:Period
        CPair(i,2+q) = randint(1,1,[1,Period]);
        end
    end


Choice=randint(1,length(Crossover)/2,[1,3]);

for i=1:length(Crossover)/2

% 1 point Crossover
if Choice(i) == 1
    Point1F = randint(1,1,[1,length(F(1,:,1))-1]);
    for q=1:Period
        for v=Point1F+1:length(F(1,:,1))
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 2 points Crossover
if Choice(i) == 2
    Point2F=randint(1,2,[1,length(F(1,:,1))-1]);
    Point2F=sort(Point2F);
    for q=1:Period
        for v=Point2F(1)+1:Point2F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 3 points Crossover
if Choice(i) == 3
    Point3F=randint(1,3,[1,length(F(1,:,1))-1]);
    Point3F=sort(Point3F);
    for q=1:Period
        for v=Point3F(1)+1:Point3F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        for w=Point3F(3)+1:length(F(1,:,1))
            PopF(CPair(i,2+q),w,CPair(i,1)) = F(CPair(i,2+q),w,CPair(i,2));
            PopF(CPair(i,2+q),w,CPair(i,2)) = F(CPair(i,2+q),w,CPair(i,1));
            end
        end
    end

end
```

A.2.3.7. Procedure MUTATION (mutation.m)

```
function [PopFM,Choice] = mutation(Par,F)

TC=Par(5);

PopFM=F;

Period=length(F(:,1,1));

Mutation=[0,0];
for s=1:Par(1)
    for v=1:Period
        RaM(s,v) = rand;
```

```
        if RaM(s,v) <= Par(4)
            Mutation = [Mutation;s,v];
            end
        end
    end
 Mutation(1,:)=[];


Choice=randint(1,length(Mutation(:,1)),[1,3]);

for u=1:length(Mutation(:,1));

% Self Mutation
if Choice(u) == 1
    TotalPoint = randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF = randint(1,TotalPoint,[1,length(F(1,:,1))]);
    A = randint(1,TotalPoint,[1,TC]);
    for i=1:TotalPoint
        while A(i) == PopFM(Mutation(u,2),PointF(i),Mutation(u,1))
            A(i)=randint(1,1,[1,TC]);
            end
        PopFM(Mutation(u,2),PointF(i),Mutation(u,1)) = A(i);
        end
    end


% Switch Mutation
if Choice(u) == 2
    P=randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        PopFM(Mutation(u,2),PointF(i,2),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,1),Mutation(u,1));
        end
    end

% Copy Mutation
if Choice(u) == 3
    P=randint(1,1,[1,round(length(F(1,:,1))/2)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        end
    end


end
```

A.2.3.8. Procedure SUMMARY (summary.m)

```
function [SolF,G,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,LC,MinC,FoL,STLI,CLI] =
summary(FOrg,P,LW,LD,M,AT,IC,C,D,T,WCC,MCC,O)

TP = length(P(1,:));
TM = length(M(1,:));
TC = length(C(1,:));
```

```
Period = length(P(:,1));


SolF=FOrg;

for v=1:Period
        for c=1:TC
            for p=1:TP
                F(v,TP*(c-1)+p,1)=0;
                if FOrg(v,p,1) == c
                   F(v,TP*(c-1)+p,1)=1;
                   end
               end
            end
        end

s=1;
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            Machine_Used = Machine_Used + P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            Worker_Used = Worker_Used + AT(m)*P(r,p)*T(p,m)*F(r,TP*(c-1)+p,s);
            end
        end
    G(r,(TM+1)*c,s) =  ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
```

```
    for m=1:TM
        L(p,m)=0;
        if T(p,m) > 0
            L(p,m)=1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                Freq = Freq + F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s);
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
Counter = 0;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        TotalT=0;
        STLI(r,TP*(c-1)+p,s)=0;
        if sum(F(r,TP*(c-1)+1:TP*c,s)) ~= 0
           FM(p,c)=.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s) ;
           STLI(r,TP*(c-1)+p,s) = LW(1,p)*F(r,TP*(c-
1)+p,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,TP*(c-
1)+p,s)*P(r,p)*sum(L(p,:))/O(1,1)/60/G(r,(TM+1)*c,s));
           end
        end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*(c-1)+1:TP*c,s);
    IndF = F(r,TP*(c-1)+1:TP*c,s);
    CLI(r,c,s) = 0;
    if sum(IndF) <= 1
       CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndF) >= 2
       CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndF)-1);
       end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      Counter = Counter + 1;
      B(c)=1;
      end
    end
  end
```

```
% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end


% Machine Change Cost
MCCost(s,r) = 0;
for c=1:TC
    for m=1:TM
        if DM(r,TM*(c-1)+m,s) < 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,2) * abs(DM(r,TM*(c-1)+m,s));
            end
        if DM(r,TM*(c-1)+m,s) > 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,1) * abs(DM(r,TM*(c-1)+m,s));
            end
        end
    end

% Worker Change Cost
WCCost(s,r) = 0;
for c=1:TC
    if DW(r,c,s) < 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,2) * abs(DW(r,c,s));
       end
    if DW(r,c,s) > 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,1) * abs(DW(r,c,s));
       end
    end


end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied FoL, & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
```

```
        if MinC(r,c,s) > C(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
           end
        if G(r,(TM+1)*c,s) > D(r,c)
           Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
           end
        if FoL(r,c,s) > 3
           Pen(s) = Pen(s) + Based_Penalty * 100 * (FoL(r,c,s) - 3);
           end
        if CLI(r,c,s) > 1.5
           Pen(s) = Pen(s) + Based_Penalty * 100 * (CLI(r,c,s) - 1.5);
           end
      end
   end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MCCost(s,:)) + sum(WCCost(s,:)) +
sum(MIC(s,:)) + sum(WIC(s,:)) + sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);
```

A.2.4. GA4 Source Codes (m-files)

A.2.4.1. Example 7: Main Algorithm (Ex7Thesis.m)

```
clear;
clc;

P = [21000,19000,20000,20000,22000,23000,22000,19000,18000,20000;
     22000,16000,18000,21000,23000, 9000,19000,20000,20000, 9000;
     21000,13000,14000,18000,19000, 6000,15000,21000,15000, 7000];
LW = [12,      9,      11,     7,      14,     16,     10,     8,      9,      12];
LD = [37,      41,     49,     54,     40,     45,     52,     36,     51,     42];
IM = [0.3,     0.5,    0.4,    0.35,   0.25,   0.4,    0.45,   0.3,    0.35,   0.4];


% Machines Information
M = [18000,20000,22000,24000,23000,19000,20000];
AT= [  .30,   .40,   .35,   .55,   .50,   .65,   .60];
IC= [    4,    6,     5,     6,     3,     4,     4];
OC= [    9,   12,    11,    12,    11,    10,     9];

% Cells Information
C = [8, 6;
     6, 6;
     6, 7];
D = [6,5;
     5,4;
     5,5];


% Parts Operations Sequence
T =    [  0,  0,1.9,  0,  0,  0,  0;    % Part1
          2,  0,  0,  0,  0,  0,  0;
          0,  0,  0,  0,1.4,  0,  0;
          0,  0,  0,1.7,  0,  0,  0;    % Part2
          0,  0,  0,  0,  0,  0,1.9;
          0,  0,  0,  0,  0,2.2,  0;
          0,  0,  0,  0,  0,  0,  2;    % Part3
          0,  0,  0,2.1,  0,  0,  0;
          0,  0,  0,  0,  0,  0,  0;
          0,  0,  0,2.3,  0,  0,  0;    % Part4
          0,1.8,  0,  0,  0,  0,  0;
          0,  0,  0,  0,  0,  0,  0;
```

```
          0,1.5,   0,   0,   0,   0,   0;     % Part5
          0,   0,  0,2.2,   0,   0,   0;
          0,   0,   0,   0,   0,   0,   0;
          0,   0,   0,   0,   0,   0,1.8;     % Part6
          0,2.5,   0,   0,   0,   0,   0;
          0,   0,   0,   0,1.9,   0,   0;
          0,   0,   0,   0,   0,1.7,   0;     % Part7
          0,   0,   0,   2,   0,   0,   0;
          0,   0,   0,   0,   0,   0,   0;
        2.1,   0,   0,   0,   0,   0,   0;     % Part8
          0,   0,2.4,   0,   0,   0,   0;
          0,   0,   0,   0,   0,   0,   0;
        1.7,   0,   0,   0,   0,   0,   0;     % Part9
          0,   0,   0,2.4,   0,   0,   0;
          0,   0,   0,   0,   0,   0,   0;
          0,   0,   0,   0,   0,1.5,   0;     % Part10
          0,2.7,   0,   0,   0,   0,   0;
          0,   0,   0,   0,1.8,   0,   0];


% Machine & Worker Change
MCC = [5000,4000;
       6000,5000;
       3000,5000];
WCC = [600,600;
       400,500;
       700,300];

% Other Information
O = [2000,10,4,20000];

% Parameter
Parameter = [30,3000,.80,.08,length(C(1,:))];

tic

[FNew] = generate(Parameter,P,M,C,T);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
       end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,IMC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O);
```

```
if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.4.2. Example 8: Main Algorithm (Ex8Thesis.m)

```
clear;
clc;

% Parts Information
P = [26000,    28000, 17000, 27000, 29000,   11000, 25000, 24000, 16000,   26000, 16000,
18000, 24000,  22000, 28000, 20000, 30000,   24000, 18000, 19000;
     22000,    16000, 18000, 21000, 23000,  9000, 19000, 20000, 20000,  9000, 13000,
15000, 10000, 16000, 14000, 22000, 24000, 20000, 19000, 22000;
     21000, 13000, 14000, 18000, 19000,  6000, 15000, 21000, 15000, 17000,  9000, 25000,
9000,  11000,  8000, 12000, 20000, 17000, 25000, 25000];
LW= [15,  9, 11,  7, 12, 17, 10,  8,  9, 12, 11, 17, 15,  6, 11, 13, 14, 21, 10,  9];
LD= [57,       41,     49,     54,     40,     45,     52,     36,     51,     42,     40,
       66,     57,     52,     43,     41,     45,     30,     37,     48];
IM = [0.3,     0.5,    0.4,    0.35,   0.25,   0.4,    0.45,   0.3,    0.35,   0.4,    0.3,
       0.5,    0.4,    0.35,   0.25,   0.35,   0.25,   0.4,    0.45,   0.5];


% Machines Information
M = [25000, 26000, 23000, 15000, 15000, 24000, 20000, 18000, 17000, 16000, 22000, 21000];
AT= [ 0.75,   0.6,     0.75,   0.7,  0.65,  0.55,     0.75,   0.7,    0.5,  0.55,   0.5,
0.55];
IC= [    4,        6,     5,     6,     3,        4,     4,     6,     5,        6,     5,
3];

% Cells Information
C = [12,       14,     12,     14;
12,     14,     11,     14;
11,     12,     9,      14];

D = [8, 8,     7,      8;
8,      8,     7,      7;
7,      7,     7,      8];



% Parts Operations Sequence
T =    [ 0,    0,      0,      0,      0,      1.8,    0,      0,      0,      0,      0,
       0;
0,     0,      1.9,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      2.1,    0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     1.2,    0,      0,      0,      0,      0,      0,      0,      0,      3,      0;
0,     0,      0,      0,      0,      2.8,    0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      2.4,    0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      2.1,    0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      2.2,    0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      2.1,    0,      0;
0,     2.1,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      2.3,    0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     1.2,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      1.9,    0,      0;
0,     0,      0,      0,      1.6,    0,      0,      0,      0,      0,      0,      0;
0,     0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
```

```
0,      0,      0,      0,      2.6,    0,      0,      0,      0,      0,      0,      0;
2.9,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      1.5,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      1.4,    0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      3.2;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      2.2,    0,      0,      0,      0;
0,      0,      1.8,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      2.5,    0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      1.8,    0,      0,      0;
0,      0,      0,      0,      0,      0,      2.8,    0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      1.7,    0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      2.1;
0,      0,      0,      0,      0,      0,      0,      2.7,    0,      0,      0,      0;
1.5,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      2.3,    0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      1.2,    0;
0,      0,      0,      0,      1.7,    0,      0,      0,      0       0,      0,      0;
0,      0,      0,      1.2,    0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      1.8,    0,      0,      0;
0,      0,      0,      0,      0,      0,      1.3,    0,      0,      0,      0,      0;
0,      2.1,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      1.6,    0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      1.3;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      1.5,    0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      2.4,    0,      0;
0,      0,      2.1,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      2.1;
0,      0,      0,      1.9,    0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      2.5,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      2.5,    0,      0,      0,      0,      0,      0;
0,      1.7,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      1.2,    0,      0,      0,      0;
2.3,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      1.3,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      3.2,    0;
0,      0,      0,      2.1,    0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      2.2,    0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      2       0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      1.4,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      2.2,    0,      0,      0,      0;
0,      0,      0,      1.6,    0,      0,      0,      0,      0,      0,      0,      0;
1.3,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      3.2,    0,      0;
1.7,    0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      2,      0,      0,      0,      0,      0,      0,      0;
0,      0,      1.5,    0,      0,      0,      0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      3.2,    0,      0,      0,      0,      0,      0;
0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      0,      1.8];


% Machine & Worker Change Cost
MCC = [4000, 5000;
       6000, 4000;
       3000, 5000;
       4000, 4500];
WCC = [600, 600;
```

```
       400, 500;
       700, 300,
       650, 400];

% Other Information
O = [2000,10,4,20000];

% Parameter
Parameter = [40,4000,.80,.08,4];


tic

[FNew] = generate(Parameter,P,M,C,T);

for g=1:Parameter(2)
    [G,Obj,Prob,Costs] = evaluation(Parameter,FNew,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O);
    [FS,Ra] = selection(Parameter,Prob,FNew);
    [FC] = crossover(Parameter,FS);
    [FM,Ch] = mutation(Parameter,FC);
    BestAveGen(g,1) = min(Obj);
    BestAveGen(g,2) = mean(Obj);
    for h=1:Parameter(1)
        if BestAveGen(g,1) == Obj(h)
           BestFGen(:,:,g) = FNew(:,:,h);
           BestGGen(:,:,g) = G(:,:,h);
           ObjGen(g) = Obj(h);
       end
    end
    FNew = FM;

end
toc

BestAll = min(BestAveGen(:,1));
for x=1:Parameter(2)
    if BestAveGen(x,1) == min(BestAveGen(:,1))
        BestF = BestFGen(:,:,x);
        BestG = BestGGen(:,:,x);
        BestCost = ObjGen(x);
    end
end

[SolF,SolG,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,IMC,LC,MinC,FoL,STLI,CLI] =
summary(BestF,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O);

if Costs(2) ~= 0
    'No Feasible Solution obtained'
else
    System_Cost
end
```

A.2.4.3. Procedure GENERATE (generate.m)

```
function [PopF] = generate(Par,P,M,C,T)

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));
Period = length(P(:,1));

for s=1:Par(1)
    PopF(:,:,s) = randint(Period,TP*TJ,[1,TC]);
end
```

A.2.4.4. Procedure EVALUATION (evaluation.m)

```
function [G,System_Cost,Prob,Costs] =
evaluation(Par,FOrg,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O)

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));

Period = length(P(:,1));

for u=1:Par(1)
  for v=1:Period
        for c=1:TC
            for p=1:TP
                for j=1:TJ
                    F(v,TP*TJ*(c-1)+TJ*(p-1)+j,u)=0;
                    if FOrg(v,TJ*(p-1)+j,u) == c
                        F(v,TP*TJ*(c-1)+TJ*(p-1)+j,u)=1;
                        end
                    end
                end
            end
        end
    end


for s=1:Par(1)
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            for j=1:TJ
                Machine_Used = Machine_Used + P(r,p)*T(TJ*(p-1)+j,m)*F(r,TP*TJ*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            for j=1:TJ
                Worker_Used = Worker_Used + AT(m)*P(r,p)*T(TJ*(p-1)+j,m)*F(r,(TP*TJ)*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        end
    G(r,(TM+1)*c,s) = ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
```

```
    end

% Intercell Moves
for c=1:TC
    for p=1:TP
        for j=2:TJ
            V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) -
F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j-1,s);
            if  (V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) == -1 )
                V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = 0;
                end
            end
        end
    end

% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
    for j=1:TJ
        L(p,j)=0;
        if sum(T(TJ*(p-1)+j,:)) > 0
            L(p,j) = 1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                for j=1:TJ
                    Freq = Freq + F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*P(r,p)*L(p,j)/O(1,1)/60/G(r,(TM+1)*c,s);
                    end
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        for j=1:TJ
```

```
        TotalT=0;
        STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        if (sum(T(TJ*(p-1)+j,:)) ~= 0) & (G(r,(TM+1)*c,s) ~=0 )
           ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = L(p,j)*F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
           FM(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = (.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = LW(1,p)*ASG(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           end
       end
       end
    end
  for c=1:TC
    IndSTLI=STLI(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    IndASG=ASG(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    CLI(r,c,s) = 0;
    if sum(IndASG) <=1
        CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndASG) >= 2
        CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndASG)-1);
        end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      B(c)=1;
      end
    end
  end


% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Intercell Movements Cost
IMC(s,r)=0;
for c=1:TC
    for p=1:TP
        for j=2:TJ
            IMC(s,r) = IMC(s,r) + P(r,p)*IM(1,p)*V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
```

```
            end
        end
    end

% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end


% Machine Change Cost
MCCost(s,r) = 0;
for c=1:TC
    for m=1:TM
        if DM(r,TM*(c-1)+m,s) < 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,2) * abs(DM(r,TM*(c-1)+m,s));
            end
        if DM(r,TM*(c-1)+m,s) > 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,1) * abs(DM(r,TM*(c-1)+m,s));
            end
        end
    end

% Worker Change Cost
WCCost(s,r) = 0;
for c=1:TC
    if DW(r,c,s) < 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,2) * abs(DW(r,c,s));
       end
    if DW(r,c,s) > 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,1) * abs(DW(r,c,s));
       end
    end


end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied Freq of Lifting & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
         if MinC(r,c,s) > C(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
            end
         if G(r,(TM+1)*c,s) > D(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
            end
         if FoL(r,c,s) > 3
            Pen(s) = Pen(s) + Based_Penalty*100* (FoL(r,c,s) - 3);
            end
         if CLI(r,c,s) > 1.5
            Pen(s) = Pen(s) + Based_Penalty*100* (CLI(r,c,s) - 1.5);
            end
        end
    end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MCCost(s,:)) + sum(WCCost(s,:)) +
sum(MIC(s,:)) + sum(WIC(s,:)) + sum(IMC(s,:)) + sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);

end  % end of pop size

Total_Fitness=0;
Dim=min(System_Cost);
```

```
for s=1:Par(1)
    Fitness_Value(s,1) = (Dim/1000) / ((Dim/1000)+System_Cost(s)-Dim);
    Total_Fitness = Total_Fitness + Fitness_Value(s,1);
    end

for s=1:Par(1)
    Prob(s,1) = Fitness_Value(s,1)/Total_Fitness;
    end

Prob(1,2)=Prob(1,1);
for s=2:Par(1)
    Prob(s,2) = Prob(s-1,2) + Prob(s,1);
    end
```

A.2.4.5. Procedure SELECTION (selection.m)

```
function [PopF,Ra] = selection(Par,Prob,F)

for s=1:Par(1)
    Ra(s) = rand;
    for t=2:Par(1)
        if Ra(s) <= Prob(1,2)
            PopF(:,:,s) = F(:,:,1);
            end
        if (Prob(t-1,2) < Ra(s)) & (Ra(s) <= Prob(t,2))
            PopF(:,:,s) = F(:,:,t);
            end
        end
    end
```

A.2.4.6. Procedure CROSSOVER (crossover.m)

```
function [PopF,RaC] = crossover(Par,F)

PopF=F;

Period=length(F(:,1,1));

Crossover=[0,0];
while mod(length(Crossover),2) ~= 1
    Crossover=0;
    for s=1:Par(1)
        RaC(s) = rand;
        if RaC(s) <= Par(3)
            Crossover = [Crossover,s];
           end
        end
    end

Crossover(1)=[];

for i=1:length(Crossover)/2
    CPair(i,1) = Crossover(2*i-1);
    CPair(i,2) = Crossover(2*i);
    for q=1:Period
        CPair(i,2+q) = randint(1,1,[1,Period]);
        end
    end


Choice=randint(1,length(Crossover)/2,[1,3]);
```

```
for i=1:length(Crossover)/2

% 1 point Crossover
if Choice(i) == 1
    Point1F = randint(1,1,[1,length(F(1,:,1))-1]);
    for q=1:Period
        for v=Point1F+1:length(F(1,:,1))
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 2 points Crossover
if Choice(i) == 2
    Point2F=randint(1,2,[1,length(F(1,:,1))-1]);
    Point2F=sort(Point2F);
    for q=1:Period
        for v=Point2F(1)+1:Point2F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        end
    end

% 3 points Crossover
if Choice(i) == 3
    Point3F=randint(1,3,[1,length(F(1,:,1))-1]);
    Point3F=sort(Point3F);
    for q=1:Period
        for v=Point3F(1)+1:Point3F(2)
            PopF(CPair(i,2+q),v,CPair(i,1)) = F(CPair(i,2+q),v,CPair(i,2));
            PopF(CPair(i,2+q),v,CPair(i,2)) = F(CPair(i,2+q),v,CPair(i,1));
            end
        for w=Point3F(3)+1:length(F(1,:,1))
            PopF(CPair(i,2+q),w,CPair(i,1)) = F(CPair(i,2+q),w,CPair(i,2));
            PopF(CPair(i,2+q),w,CPair(i,2)) = F(CPair(i,2+q),w,CPair(i,1));
            end
        end
    end

end
```

A.2.4.7. Procedure MUTATION (mutation.m)

```
function [PopFM,Choice] = mutation(Par,F)

TC=Par(5);

PopFM=F;

Period=length(F(:,1,1));

Mutation=[0,0];
for s=1:Par(1)
    for v=1:Period
        RaM(s,v) = rand;
        if RaM(s,v) <= Par(4)
            Mutation = [Mutation;s,v];
            end
        end
    end
 Mutation(1,:)=[];
```

```
Choice=randint(1,length(Mutation(:,1)),[1,3]);

for u=1:length(Mutation(:,1));

% Self Mutation
if Choice(u) == 1
    TotalPoint = randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF = randint(1,TotalPoint,[1,length(F(1,:,1))]);
    A = randint(1,TotalPoint,[1,TC]);
    for i=1:TotalPoint
        while A(i) == PopFM(Mutation(u,2),PointF(i),Mutation(u,1))
            A(i)=randint(1,1,[1,TC]);
            end
        PopFM(Mutation(u,2),PointF(i),Mutation(u,1)) = A(i);
        end
    end


% Switch Mutation
if Choice(u) == 2
    P=randint(1,1,[1,round(length(F(1,:,1))/3)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        PopFM(Mutation(u,2),PointF(i,2),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,1),Mutation(u,1));
        end
    end

% Copy Mutation
if Choice(u) == 3
    P=randint(1,1,[1,round(length(F(1,:,1))/2)]);
    PointF=randint(P,2,[1,length(F(1,:,1))]);
    for i=1:P
        while PointF(i,1) == PointF(i,2)
            PointF(i,2) = randint(1,1,[1,length(F(1,:,1))]);
            end
        PopFM(Mutation(u,2),PointF(i,1),Mutation(u,1)) =
F(Mutation(u,2),PointF(i,2),Mutation(u,1));
        end
    end


end
```

A.2.4.8. Procedure SUMMARY (summary.m)

```
function [SolF,G,System_Cost,Costs,MC,WC,MIC,WIC,MCCost,WCCost,IMC,LC,MinC,FoL,STLI,CLI]
= summary(FOrg,P,LW,LD,IM,M,AT,IC,C,D,T,WCC,MCC,O)

SolF=FOrg;

TP = length(P(1,:));
TJ = length(T(:,1))/TP;
TM = length(M(1,:));
TC = length(C(1,:));

Period = length(P(:,1));

    for v=1:Period
```

```
        for c=1:TC
            for p=1:TP
                for j=1:TJ
                    F(v,TP*TJ*(c-1)+TJ*(p-1)+j,1)=0;
                    if FOrg(v,TJ*(p-1)+j,1) == c
                        F(v,TP*TJ*(c-1)+TJ*(p-1)+j,1)=1;
                        end
                    end
                end
            end
        end



s=1;
for r=1:Period

% CONSTRAINTS
% Machine Capacity
for c=1:TC
    for m=1:TM
        Machine_Used=0;
        for p=1:TP
            for j=1:TJ
                Machine_Used = Machine_Used + P(r,p)*T(TJ*(p-1)+j,m)*F(r,TP*TJ*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        G(r,(TM+1)*(c-1)+m,s) = ceil(Machine_Used/60/O(1,1));
        MI(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s)*O(1,1)*60 - Machine_Used;
        end
    end

% Cell Capacity
for c=1:TC
    MinC(r,c,s)=0;
    for m=1:TM
        MinC(r,c,s) = MinC(r,c,s) + G(r,(TM+1)*(c-1)+m,s);
        end
    end

% Worker Capacity
for c=1:TC
    Worker_Used=0;
    for m=1:TM
        for p=1:TP
            for j=1:TJ
                Worker_Used = Worker_Used + AT(m)*P(r,p)*T(TJ*(p-1)+j,m)*F(r,(TP*TJ)*(c-
1)+TJ*(p-1)+j,s);
                end
            end
        end
    G(r,(TM+1)*c,s) = ceil(Worker_Used/60/O(1,1));
    WI(r,c,s) = G(r,(TM+1)*c,s)*O(1,1)*60 - Worker_Used;
    end

% Intercell Moves
for c=1:TC
    for p=1:TP
        for j=2:TJ
            V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) -
F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j-1,s);
            if  (V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) == -1 )
                V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = 0;
                end
            end
        end
    end
```

```
% Machine & Worker Balance Equation
for c=1:TC
    for m=1:TM
        DM(r,(TM+1)*(c-1)+m,s)=0;
        if r > 1
            DM(r,TM*(c-1)+m,s) = G(r,(TM+1)*(c-1)+m,s) - G((r-1),(TM+1)*(c-1)+m,s);
            end
        end
    DW(r,(TM+1)*c,s)=0;
    if r > 1
        DW(r,c,s) = G(r,(TM+1)*c,s) - G((r-1),(TM+1)*c,s);
        end
    end


% Frequency of Lifting
for p=1:TP
    for j=1:TJ
        L(p,j)=0;
        if sum(T(TJ*(p-1)+j,:)) > 0
            L(p,j) = 1;
            end
        end
    end
for c=1:TC
    A=1;
    FoL(r,c,s) = 0;
    if G(r,(TM+1)*c,s) ~= 0
        while A == 1
            Freq = 0;
            for p=1:TP
                for j=1:TJ
                    Freq = Freq + F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*P(r,p)*L(p,j)/O(1,1)/60/G(r,(TM+1)*c,s);
                    end
                end
            FoL(r,c,s) = Freq;
            A=0;
            if (FoL(r,c,s) > 3) & (G(r,(TM+1)*c,s) <= D(r,c))
                G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
                WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
                A=1;
                end
            end
         end
     end


% Composite Lifting Index
B(1:TC)=1;
while max(B) == 1
  for c=1:TC
    for p=1:TP
        for j=1:TJ
        TotalT=0;
        STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s)=0;
        if (sum(T(TJ*(p-1)+j,:)) ~= 0) & (G(r,(TM+1)*c,s) ~=0 )
           ASG(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = L(p,j)*F(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
           FM(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = (.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           STLI(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s) = LW(1,p)*ASG(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)/19.55/(0.82+4.5/LD(1,p))/(.8359-.0893464*F(r,(TP*TJ)*(c-1)+TJ*(p-
1)+j,s)*L(p,j)*P(r,p)/O(1,1)/60/G(r,(TM+1)*c,s));
           end
       end
       end
    end
  for c=1:TC
```

```
    IndSTLI=STLI(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    IndASG=ASG(r,TP*TJ*(c-1)+1:TP*TJ*c,s);
    CLI(r,c,s) = 0;
    if sum(IndASG) <=1
        CLI(r,c,s) = max(IndSTLI);
        end
    if sum(IndASG) >= 2
        CLI(r,c,s) = max(IndSTLI) + 0.25*(sum(IndSTLI)-max(IndSTLI))/(sum(IndASG)-1);
        end
    B(c)=0;
    if (CLI(r,c,s) > 1.5) & (G(r,(TM+1)*c,s) <= D(r,c))
      G(r,(TM+1)*c,s) = G(r,(TM+1)*c,s) + 1;
      WI(r,c,s) = WI(r,c,s)+O(1,1)*60;
      B(c)=1;
      end
    end
  end


% OBJECTIVE FUNCTION
% Machine Cost
MC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MC(s,r) = MC(s,r) + M(m) * G(r,(TM+1)*(c-1)+m,s) ;
        end
    end

% Machine Idle Cost
MIC(s,r) = 0;
for c=1:TC
    for m=1:TM
        MIC(s,r) = MIC(s,r) + IC(m) * MI(r,TM*(c-1)+m,s) / 60;
        end
    end

% Worker Cost
WC(s,r)=0;
for c=1:TC
    WC(s,r) = WC(s,r) + O(1,2)*O(1,1)*G(r,(TM+1)*c,s);
    end

% Worker Idle Cost
WIC(s,r)=0;
for c=1:TC
    WIC(s,r) = WIC(s,r) + O(1,3)*WI(r,c,s)/60;
    end

% Intercell Movements Cost
IMC(s,r)=0;
for c=1:TC
    for p=1:TP
        for j=2:TJ
            IMC(s,r) = IMC(s,r) + P(r,p)*IM(1,p)*V(r,(TP*TJ)*(c-1)+TJ*(p-1)+j,s);
            end
        end
    end

% Lifting Cost
LC(s,r)=0;
for c=1:TC
    LC(s,r) = LC(s,r) + O(1,4)*G(r,(TM+1)*c,s)*CLI(r,c,s);
    end


% Machine Change Cost
MCCost(s,r) = 0;
for c=1:TC
    for m=1:TM
```

```
        if DM(r,TM*(c-1)+m,s) < 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,2) * abs(DM(r,TM*(c-1)+m,s));
            end
        if DM(r,TM*(c-1)+m,s) > 0
            MCCost(s,r) = MCCost(s,r) + MCC(r,1) * abs(DM(r,TM*(c-1)+m,s));
            end
        end
    end

% Worker Change Cost
WCCost(s,r) = 0;
for c=1:TC
    if DW(r,c,s) < 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,2) * abs(DW(r,c,s));
       end
    if DW(r,c,s) > 0
       WCCost(s,r) = WCCost(s,r) + WCC(r,1) * abs(DW(r,c,s));
       end
    end


end   % end of period

Based_Penalty = max(MC(s,:));

% Penalty for unsatisfied Cell Capacity, unsatisfied Freq of Lifting & unsatisfied CLI
Pen(s)=0;
for r=1:Period
    for c=1:TC
         if MinC(r,c,s) > C(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (MinC(r,c,s) - C(r,c));
            end
         if G(r,(TM+1)*c,s) > D(r,c)
            Pen(s) = Pen(s) + Based_Penalty * (G(r,(TM+1)*c,s) - D(r,c));
            end
         if FoL(r,c,s) > 3
            Pen(s) = Pen(s) + Based_Penalty*100* (FoL(r,c,s) - 3);
            end
         if CLI(r,c,s) > 1.5
            Pen(s) = Pen(s) + Based_Penalty*100* (CLI(r,c,s) - 1.5);
            end
        end
    end

Costs(s,1) = sum(MC(s,:)) + sum(WC(s,:)) + sum(MCCost(s,:)) + sum(WCCost(s,:)) +
sum(MIC(s,:)) + sum(WIC(s,:)) + sum(IMC(s,:)) + sum(LC(s,:)) ;
Costs(s,2) = Pen(s);
System_Cost(s) = Costs(s,1) + Costs(s,2);
```

Vita Auctoris

Setiadi Lesmana was born November 17, 1980 in Tarakan, Indonesia. He attended University of Surabaya, Surabaya, Indonesia from 1996 to 2000 and graduated with a Bachelor of Engineering in Industrial Engineering. He is currently a candidate for the Master of Applied Science in Industrial and Manufacturing Systems Engineering at the University of Windsor and hopes to graduate in December 2002.